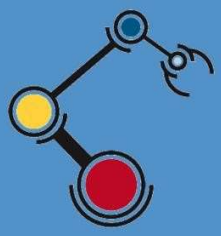
Escuela Politécnica Superior

Interfaz de usuario de voz para sistemas de Automatización



Máster Universitario en Automática y Robótica

Trabajo Fin de Máster

Autor:

Ilias Diamantaras

Tutor/es:

Francisco A. Candelas Herías

Julio 2019

Universitat d'Alacant
Universidad de Alicante

Universidad de Alicante

Máster Universitario en Automática y Robótica

Trabajo de Fin de Máster

Interfaz de usuario de voz para sistemas de automatización

Autor

Ilias Diamantaras

Tutores

Francisco A. Candelas Herías

Alicante, Julio de 2019

Dedicatoria

A María por todo su apoyo y paciencia todo este tiempo.

A mi Familia y mis amigos por animarme y apoyarme cuando lo necesito.

Agradecimientos

Agradezco mi tutor Francisco por ayudarme y encaminarme a la resolución de los problemas encontrados. A Alex por ayudarme con Python. A Valentín por ayudarme en el trabajo. Con vuestra ayuda ha sido posible desarrollar este proyecto.

JUSTIFICACION Y OBJETIVOS

En los últimos años los asistentes virtuales o de voz incluidos en el software de dispositivos móviles han experimentado y han extendido más su utilización. Además, han surgido dispositivos asistentes, económicos, que facilitan una interacción más intuitiva con otros dispositivos electrónicos de la vida cotidiana, y con servicios de internet. En contraste, los equipos y aplicaciones para interfaces de usuario avanzadas que se utilizan en automatización industrial se basan principalmente en pantallas gráficas táctiles HMI, porque son más adecuadas y robustas en las aplicaciones industriales en entornos de condiciones duras. Pero el equipamiento de automatización, incluyendo los PLCs (controladores lógicos programables) también se utiliza para aplicaciones en entornos no industriales o menos duros, como por ejemplo la automatización de edificios o viviendas, conviviendo con los sistemas específicos de domótica. En estas aplicaciones, puede resultar muy interesante complementar los controladores con otros tipos de interfaz de usuario no gráficos, como por ejemplo los basados en voz y diálogos. En este sentido, el trabajo de propuesta abordará la interconexión de un asistente de voz comercial con un sistema de automatización gobernado por PLC, para que una persona pueda interactuar con el sistema mediante Comandos de Voz.

Motivación del proyecto.

El principal motivo que ha llevado a la idea de este trabajo de final de máster, consiste en el interés por una interfaz de voz hombre-máquina o Interfaz voz de usuario (IVU) que se pueda utilizar con cualquier autómata. Además, existe un interés de todo tipo de automatización en general. Normalmente en el mundo industrial no se ha tenido en cuenta un sistema de interfaz de voz hombre-máquina. El uso de un sistema que permite al usuario poder interactuar con la máquina o sistema central de domótica, de una manera natural, permite desarrollar interfaces intuitivas que no se ha podido hasta ahora.

Los objetivos generales son los siguientes. Primero, la selección de un asistente de voz que se pueda desarrollar una interfaz de alto nivel. Segundo, la selección de un PLC de Siemens que se pueda utilizar en conjunto con la Raspberry Pi. Tercero, utilizar los protocolos necesarios

para la comunicación entre todos los elementos de la aplicación. Cuarto, programar el asistente de voz Amazon Alexa, PLC y Raspberry Pi para el funcionamiento del sistema.

En este proyecto se ha fabricado un cuadro eléctrico para poder realizar las pruebas con todos los elementos conectados entre ellos. También se ha utilizado para demostrar las posibilidades que tiene una interfaz de voz conectada con un PLC.

GROUNDS AND OBJECTIVES

In recent years, virtual or voice assistants included in mobile device software have undergone a development and their use has been further extended. In addition, assistant, economic devices have emerged, which facilitate a more intuitive interaction with other electronic devices in the daily life, with internet services. In contrast, the equipment and applications for advanced user interfaces used in industrial automation are based mainly on Human machine interface HMI, because they are more suitable and robust in industrial applications in harsh environments. The automation hardware, including PLCs (programmable logic controllers) is also used for applications in non-industrial or less harsh environments, such as the automation of buildings or homes, coexisting with specific systems of home automation. In these applications, it can be very interesting to complement the Hardware with other non-graphic user interface types, such as those based on voice and dialogs. The proposed work will address the interconnection of a commercial voice assistant with an automation system controlled by a PLC, so that a person can interact with the system through Voice commands.

MOTIVATION FOR THE PROJECT

The main reason that led to the idea of the thesis, is the interest in a human-machine voice interface or user voice interface (UVI) that can be used with any PLC. In general, the author has an interest that stems from anything related to automation. In the industrial world, a human-machine voice interface system has not been usually considered. The use of a system that allows the user to interact with the central machine or home automation system, in a natural way, allows intuitive interfaces to be developed that have not been possible until now. The general

objectives are the following. First, the selection of a voice assistant that can develop a high-level interface. Second, the selection of a Siemens PLC that can be used in conjunction with the Raspberry Pi. Third, use the necessary protocols for communication between all the elements of the application. Fourth, program the voice assistant Amazon Alexa, PLC and Raspberry Pi for the correct operation of the system.

In this project an electrical panel has been manufactured to be able to carry out the tests with all the elements connected between them. It has also been used to demonstrate the capabilities of a voice interface connected to a PLC.

Palabras clave

Amazon Alexa, Raspberry Pi, Interfaz de usuario de voz (IVU), PLC, Siemens, Automatización.

Keywords.

Amazon Alexa, Raspberry Pi, Voice user interface “VUI”, PLC, Siemens, Automation.

Índice de Contenidos

Índice de abreviaturas

IVU: Interfaz mediante voz del usuario

HMI: Human machine interface

EPS: Escuela Politécnica Superior

RAH: Reconocimiento automático del habla

PLC: Programable Logic Controller

ASK: Alexa Skills Kit

API: Application Programming Interface

JSON: Java Script Object Notation

PIXEL: Pi Improved x Windows Environment, Lightweight

VNC: Virtual Network Computing

ST: Structured Text

TIA Portal: Totally integrated Automation Portal

1. INTRODUCCION

1.1 Visión General

Las interfaces hombre-máquina mediante voz del usuario permiten la interacción con ordenadores a través de una plataforma como Amazon Echo para iniciar procesos o servicios automatizados. IVU (o Interfaz mediante voz) es la interfaz de cualquier aplicación de habla. El uso de interfaces bien diseñadas permite a los usuarios poder trabajar de manera más rápida y simplificada. El reconocimiento de Voz y las interfaces han evolucionado mucho y el cambio en este sector ha sido muy grande.

En este proyecto se ha propuesto un nuevo desarrollo de interfaz de voz mediante la plataforma de Amazon Alexa conectada a un PLC siemens a través la Raspberry Pi. El proyecto está basado en la idea de que se simula una máquina con un PLC. El PLC comunica con la Raspberry Pi mediante el protocolo Modbus. Dentro de la Raspberry Pi se utilizarán varias bibliotecas y programas, para enlazar el PLC que simula una máquina, con el altavoz Echo. Cuanto más cerca se encuentre la IVU de las necesidades del usuario final para esta tarea, más fácil será de utilizar y el coste de tiempo será menor, resultando en mayor eficiencia y satisfacción del usuario.

No todos los procesos de automatización se desenvuelven igualmente bien con la automatización basado en la IVU. Generalmente, cuanto más complejas sean las interacciones, más complicado serán de automatizar y más fallará con el usuario final. En algunos casos, la automatización no se puede aplicar, así que una interfaz estándar tipo HMI es la única opción. Por ejemplo, una línea de producción sería muy complicada de automatizar con IVU. Por otro lado, la interfaz con IVU es perfecta para consultar el estado o la producción de una máquina de forma rápida. Consultar que equipos están activados o el estado de la iluminaria de una sala de trabajo.

Actualmente existen varias plataformas de IVU, entre las que destacan, es la de Amazon (Alexa), Google (OK Google) y Apple (Siri), que permiten utilizarlas con varios dispositivos. Algunas de estas plataformas permiten integrar varios dispositivos a través de API que ofrecen para los desarrolladores. Una de ellas es Alexa que utiliza la familia de altavoces Echo de la compañía Amazon. Como se ha mencionado anteriormente, las características y el diseño de la IVU son muy importantes. Alexa es el asistente de voz, y Echo es uno de los dispositivos de Amazon que lo utilizan. Además de Echo, Amazon ofrece otros dispositivos como Echo Plus, Echo Dot. Alexa en realidad es el nombre de la tecnología de inteligencia artificial, no del producto.

1.2. Estructura de los contenidos

Una vez explicado e introducido el trabajo final de master, en el siguiente capítulo se realizará una breve introducción del proyecto.

Después, en el capítulo 2 del Estado del Arte, se presentan la información y las tecnologías utilizadas para poder reproducir el trabajo desarrollado. Tras esto, en el capítulo 3 de Desarrollo, se expone el trabajo llevado a cabo para poner en marcha los diferentes sistemas y conceptos teóricos y prácticos de la interfaz IVU. Se expondrán los diferentes problemas que han surgido durante su programación y las decisiones de diseño que se han tomado para solventarlos.

Finalmente, en el capítulo 4 sobre Pruebas, se muestra una serie de experimentos, para validar el sistema desarrollado, y después, en el capítulo 5, las conclusiones junto con los trabajos futuros y cómo se podrían implementar estos trabajos futuros.

Al final del documento se añaden los anexos en los que se puede ver los esquemas de montaje y esquemas eléctricos de la placa montada.

2. Estado del Arte

2.1 Raspberry Pi

2.1.1 Sistema Operativo

Raspbian es el sistema operativo más importante de la Raspberry Pi, una computadora de ultra bajo coste. Raspbian es el sistema operativo recomendado para Raspberry Pi (al estar optimizado para su hardware) y se basa en una distribución de GNU/Linux llamada Debian. Desde 2015, la Fundación Raspberry Pi lo ha proporcionado oficialmente como el sistema operativo principal para la familia de computadoras de una sola placa Raspberry Pi. Raspbian está altamente optimizado para las CPU ARM de bajo rendimiento de la línea Raspberry Pi.

Raspbian ofrece más que un sistema operativo puro: viene con más de 35,000 paquetes, software pre-compilado en un formato agradable para una fácil instalación en una Raspberry Pi. La compilación inicial es de más de 35,000 paquetes de Raspbian, optimizada para el mejor rendimiento en la Raspberry Pi, se completó en junio de 2012. Sin embargo, Raspbian todavía está en desarrollo activo con un énfasis en mejorar la estabilidad y el rendimiento de la mayor cantidad posible de paquetes de Debian. Raspbian utiliza PIXEL (Pi Improved x Windows Environment, Lightweight), un entorno X-Window mejorado específicamente para Raspberry Pi, como entorno de escritorio principal a partir de la última actualización.

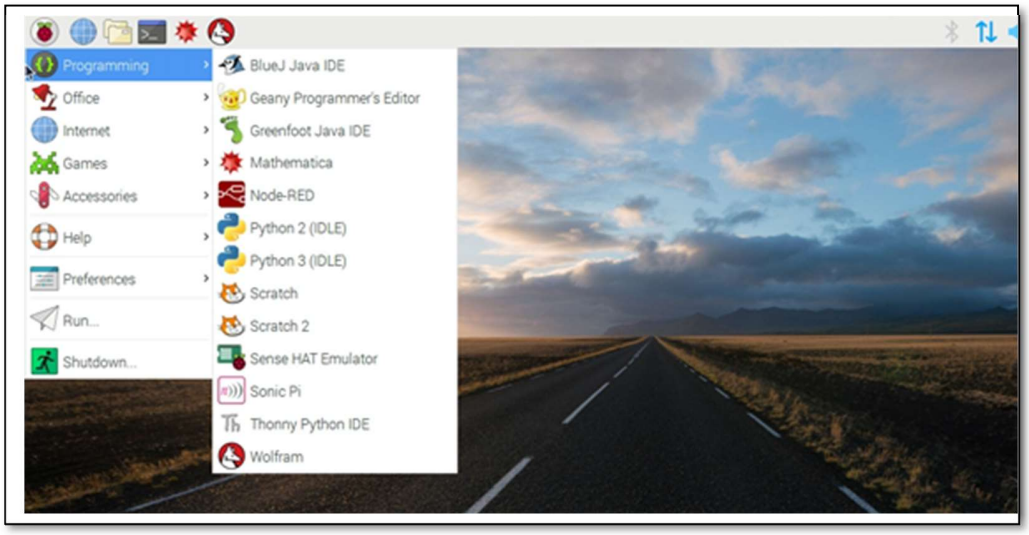


Figura 1: Sistema operativo Rasbian.

2.1.2 Instalación Sistema Operativo

Raspberry tiene varias versiones que podemos utilizar. Se ha elegido Raspbian por la facilidad de uso y la multitud de librerías que tiene.

Antes de instalar Raspbian hay que formatear la tarjeta SD con formato FAT32. Se va a utilizar NOOBS (New out of the box software), es un administrador de instalación de sistema operativo fácil para la Raspberry Pi. Hay varias descargas disponibles, se ha elegido Raspbian.

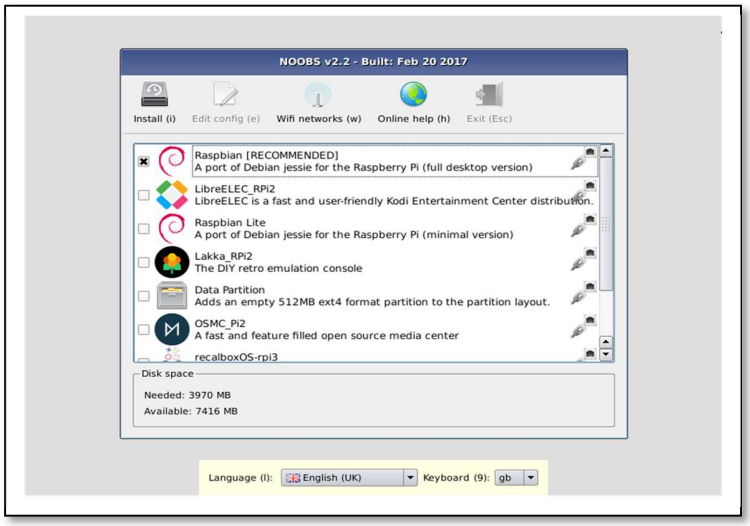


Figura 2: Selección de sistema operativo.

Una vez instalado el sistema operativo debemos configurar la Raspberry pi con la Red inalámbrica y local. En este proyecto es imprescindible tener un acceso inalámbrico a internet y local con él PLC. El punto de acceso inalámbrico (Wi-Fi) se utiliza para comunicar por HTTPS con Alexa. El puerto ethernet se utiliza para comunicar localmente con el protocolo Modbus TCP/IP con el PLC S7-1200 de siemens.

A continuación, es importante activar el servidor VNC (Virtual Network Computing o Computación Virtual en Red) que es un sistema de comunicación remota basado en el protocolo Remote FrameBuffer. VNC nos permite transmitir la señal de video y los eventos de ratón y

teclado en la red local. De esta forma se puede acceder en la Raspberry y hacer los ajustes necesarios sin una pantalla ratón y teclado. Solo será necesario el portátil que tiene todos los programas necesarios para acceder todo el hardware relacionado con el proyecto.

2.1.3 Protocolos de Comunicación

Los protocolos de comunicación que se pueden utilizar con un PLC son una parte crítica hoy en día, las máquinas se conectan cada vez más entre ellas y con otros sistemas. Los protocolos de comunicación son un conjunto de reglas y pasos estrictos, realizados automáticamente por un dispositivo de comunicaciones para establecer una conexión e intercambiar datos. Esto permite la integración de distintos equipos que normalmente no se podrían comunicar entre ellos. Los protocolos de comunicación simplifican enormemente la instalación y operación de varias máquinas.

En el mercado hay muchos protocolos de comunicación. Los más populares y ampliamente en aplicaciones industriales utilizados son los estándares sin duda reconocidos como Profibus DP, Profinet-IO, Modbus RTU, Modbus / TCP, Ethercat o EtherNet / IP. La elección de uno de ellos depende principalmente del proveedor del sistema de control de PLC, los parámetros técnicos del protocolo y su precio. En esto proyecto se han considerado dos estándares: Los protocolos basados en TCP/IP y protocolos de buses de campo o “Fieldbus”. Con los protocolos serie, la transmisión de datos normalmente se implementa en la tecnología RS-485. Una tecnología bastante vieja que limita la cantidad de datos que se pueden transmitir entre equipos, aunque se sigue usando bastante porque permite conexiones cableadas sencillas para distancias muy largas.

TCP / IP es un conjunto de protocolos de comunicación desarrollados en la década de 1970 por la Agencia de Proyectos de Investigación Avanzada de la Defensa (DARPA) del Departamento de Defensa de los Estados Unidos. Fue desarrollado en la red de DARPA llamada ARPANET y evolucionó para ser utilizado para la comunicación en internet. Hoy en día, todos los ordenadores que se conectan a internet o redes locales o internet utilizan estos protocolos.

Actualmente, Modbus es un estándar adoptado por la mayoría de los fabricantes de controladores industriales, para la comunicación asíncrona entre dispositivos equipados con una interfaz compatible con las interfaces RS-232, RS-422 o RS-485, así como con interfaces Ethernet sobre las que se usan los protocolos TCP/IP. El protocolo Modbus sigue presente en los controladores actuales de todos los fabricantes, Siemens, Allen Bradley etc. La popularidad y prevalencia de Modbus se basa en características tales como la facilidad de uso (tanto en la implementación como en la operación) y bajo coste.

Una vez desarrollado el protocolo de comunicación Modbus en la Raspberry Pi, esta se podrá comunicar con cualquier equipo del mercado que soporte Modbus. Además, no requiere ningún cambio o nuevo desarrollo si se cambia el equipo que se comunica. El coste de la implementación basada en Modbus, es muy baja, uno de los principales motivos de elegir este protocolo de comunicación.

En este proyecto se ha considerado el PLC que se va a utilizar y que tipo de protocolos es capaz de usar y el coste que tiene cada uno de ellos. El PLC escogido es el Siemens S7-1200 de la marca Siemens, es el principal fabricante en Europa y uno de los más grandes del mundo. El autómata S7-1200 está dentro de la nueva familia de gama media de Siemens, y, dentro de esta gama, es el más utilizado actualmente. Los protocolos que podemos utilizar sin coste con este PLC, son S7comm y Modbus TCP/IP. Utilizando la librería Snap7 también es posible comunicar con los PLC de Siemens en modo cliente y servidor utilizando la librería Snap7. Por otro lado, utilizando la librería Pymodbus es posible comunicar con cualquier PLC como cliente o servidor.

2.1.4 Pymodbus y Modbus

Pymodbus es una implementación del protocolo Modbus mediante twisted/asyncio/tornado. Su objetivo original era permitir la simulación de varios dispositivos Modbus en una sola máquina o dispositivo para monitorización de pruebas de software. El desarrollo de Pymodbus empezó el junio de 2009 para los sistemas operativos Linux y Unix. La librería de Pymodbus utiliza el lenguaje Python. Las versiones que podemos utilizar son 2.7 y 3+.

Modbus tiene varias funciones que se distinguen con el número de código que tiene cada una de ellas. Los códigos de función más utilizados son los siguientes.

Código de función 1: Read Coils (Leer salidas).

Código de función 2: Read Discrete Input (Leer entradas discretas).

Código de función 3: Read Holding Register (Leer registro).

Código de función 4: Read Input Register (Leer registro).

Código de función 5: Force Single Coild (Forzar Salida).

Código de función 6: Write Single Register (Escribir registro).

Pymodbus puede comunicar como cliente o servidor y cada caso tiene distintas características. Las características que tiene como Cliente:

- Protocolo completo de lectura, escritura en discreto y registro.
- La mayoría del protocolo (diagnóstico, archivo, tubería, configuración, información.
- TCP, UDP, ASCII en serie, RTU en serie y Binario en serie.
- Asíncrono (alimentado por twisted, tornado, asyncio) y versiones síncronas.

Las características que tiene como Servidor:

- Puede funcionar como un servidor Modbus completamente implementado.
- TCP, UDP, ASCII en serie, RTU en serie y Binario en serie.
- Asíncrono (Twisted) y versiones síncronas.
- Control completo del servidor (información del dispositivo, contadores, etc.).

2.1.5 Ngrok

Ngrok es un software multiplataforma de túnel y proxy inverso que establece túneles seguros desde un "Endpoint" público, como internet, a un servicio de red que se ejecuta localmente mientras que captura todo el tráfico para una inspección y reproducción.

Los posibles usos de esta herramienta son:

- Realizar test de los cambios hechos en pre-producción.
- Realizar pruebas para conocer cómo se muestran las webs en distintos navegadores.
- Mostrar avances en el desarrollo del proyecto, de manera que el cliente pueda ver los cambios solicitados y el estado del desarrollo.
- Realizar test de seguridad para conocer si las configuraciones son adecuadas.

Ngrok permite esto sin hacer ninguna configuración extra en el Firewall o en el Router. Descargando una aplicación en la Raspberry se puede utilizar Ngrok sin ningún ajuste adicional.

En la versión gratuita, Ngrok crea una nueva URL en un servidor de internet cada vez que se ejecuta la aplicación en un equipo local privado, como por ejemplo la siguiente:

https://34dda1d8.ngrok.io/

A esta URL pública se puede conectar cualquier equipo que quiera conectarse al servidor privado, para intercambiar con él información por un túnel seguro. En el proyecto se va a utilizar la URL para hacer un túnel seguro con la Aplicación de Amazon Alexa.

2.1.6 Flask-Ask

Flask-Ask es una extensión de Flask para escribir "skills" de Alexa para Amazon Echo. Los "skills" de Flask-Ask con Alexa son interacciones de voz o conversaciones con dispositivos Echo. Por ejemplo, puede escribir un "skill" para preguntar al Eco "What day is it today" y el Eco puede

responder "Today its Monday". Es posible que haya conversaciones más complejas que incluyan preguntas y el procesamiento del input del dispositivo Echo.

Flask se clasifica como un “microframework” porque no requiere herramientas o bibliotecas particulares. No tiene una capa de base de datos, validación de formularios ni ningún otro componente en el que las bibliotecas de terceros proporcionen funciones comunes. Sin embargo, Flask admite extensiones que pueden agregar características de la aplicación como si estuvieran implementadas en el mismo Flask. Existen varias extensiones basadas en Flask y una de ellas es Flask-Ask. Flask fue creado por Armin Ronarcher de pocoo, un grupo de entusiastas de Python formado en 2004. Está basado en la especificación WSGI de Werkzeug y el motor de “templates” Jinja2

Con Flask-Ask es posible:

- Mapear solicitudes y “intent slots”.
- Verificar solicitudes de Alexa.
- Permitir la separación de código y voz a través de plantillas Jinja.
- Facilitar la gestión de sesiones.
- Ayudar a construir respuestas.

Además, con Flask-Ask se puede utilizar varios API de Amazon como “AMAZON.DATE”, “AMAZON.TIME” y “AMAZON.DURATION”. Esto facilita la programación y limita el tiempo a programar solamente la aplicación y los “skills” de Alexa.

2.1.7 Sqlite

SQLite es un sistema de gestión de bases de datos relacional. Implementa un motor de base de datos sin necesidad de un servidor o necesidad de configuración. SQLite lee y escribe directamente en archivos de disco. Una base de datos completa con múltiples tablas, índices subrutinas y funciones está contenida en un solo archivo. Es una biblioteca compacta, su tamaño con todas las funciones activadas no supera los 600kb.

La biblioteca puede ser utilizada en varios lenguajes de programación, una de ellas es Python. Python incluye soporte para SQLite nativamente desde la versión 2.5 incorporado en la Biblioteca Estándar como el módulo sqlite3.

Las razones de utilizar SQLite principalmente son que tiene un tamaño pequeño. Realiza las operaciones de manera eficiente y se puede utilizar a Hardware de bajo rendimiento como Raspberry Pi. La base de datos de SQLite puede ser portada sin tener que copiar configuración adicional.

2.1.7 Python-Schedule

El proceso de rellenar la base de datos SQLite es un proceso periódico a intervalos. Schedule es un proceso para trabajos periódicos. Permite ejecutar funciones de Python, o cualquier otra opción.

Teniendo este proceso ejecutando en intervalos predeterminados es posible rellenar la base de datos sin ningún control externo.

2.2 PLC Siemens

2.1.2 PLC

Los PLC conocidos como “Programable Logic Controller” o “Procesador Lógico Programable”, conocidos también como autómatas, son ordenadores industriales muy utilizados en la automatización industrial. Están diseñados para múltiples señales de entradas y salidas, resistencia contra los ruidos eléctricos y vibración.

Estos autómatas se utilizan en casi todos los sectores de la industria, y en todo tipo de aplicaciones en procesos y secuencias de maquinaria. Hoy en día el sistema PLC puede abarcar varios procesos y uno de ellos es conectar a internet, ser monitorizado remotamente y mandar datos si fuese necesario.

En este proyecto se simula una máquina que utiliza un PLC con producción diaria. El PLC que se utiliza es de Siemens, el modelo es S7-1200. S7-1200 pertenece en la nueva familia de PLCs de gama media que ha lanzado Siemens. Se puede encontrar en multitudes de aplicaciones.

2.2.2 TIA PORTAL

TIA Portal "Totally integrated Automation Portal" es un sistema de ingeniería que permite configurar todos los procesos relacionados con los PLC de Siemens. El "framework" de ingeniería común en el que están integrados los productos de software unifica todas las funciones comunes, incluso en lo relativo a su representación en la pantalla.

Un programa desarrollado con TIA Portal este compuesto por diferentes capas de programa. Se divide entre PLC, periferias (HMI), variadores, etc.

2.3 Amazon Alexa

2.3.1 Reconocimiento de Voz

En los últimos años, varias empresas como Amazon, Google y centros de investigación de todo el mundo han hecho un gran esfuerzo avanzar el Reconocimiento Automático de Habla o RAH. Gracias a estos esfuerzos Reconocimiento Automático de Habla o RAH se ha convertido a lo que es hoy en día.

Los orígenes de RAH se encuentran en los años 1950 y 1960. El primer sistema de reconocimiento de voz pudo entender solo dígitos. El 1952 los tres investigadores Stephen Balashek, R. Biddulph y K. H. Davis construyeron un sistema llamado "Audrey" para el reconocimiento de dígitos de un solo altavoz. Diez años después IBM demostró la capacidad de reconocimiento de voz de la máquina "Shoebox" de 16 palabras en la Feria Mundial de 1962. Los laboratorios de los Estados Unidos, Japón, Inglaterra y la Unión Soviética, expandieron la tecnología de reconocimiento de voz.

El reconocimiento de voz dio grandes pasos en la década de 1970 gracias a DARPA. Financiaron durante cinco años la investigación de comprensión del habla, la investigación de reconocimiento de voz que buscaba un vocabulario mínimo de 1.000 palabras. En la década de los 80 gracias a nuevas técnicas, enfoques para la compresión de lo que dice la gente y las capacidades de los ordenadores que aumentaban rápidamente, se progresó mucho.

El rendimiento de los sistemas de Voz generalmente se evalúa en términos de precisión y velocidad. La precisión se suele calificar con una tasa de error de palabra, mientras que la

velocidad se mide con el factor de tiempo real. Otras medidas de precisión incluyen la tasa de error de una sola palabra y la tasa de éxito de comando.

Sin embargo, el reconocimiento de voz por la máquina es un problema muy complejo. Las vocalizaciones varían en términos de acento, pronunciación articulación, nasalidad, tono, volumen velocidad. La voz está distorsionada por un ruido de fondo y ecos, características eléctricas. La precisión de reconocimiento de voz puede variar con el tamaño de vocabulario, habla discontinua o continua, condiciones adversas.

Actualmente la investigación se concentra, por un lado, en mejorar el rendimiento de la modelización acústica y por otro lado en la integración de niveles de conocimiento de voz.

2.3.2 Reconocimiento de Voz de Amazon

Alexa, nombrado de la antigua biblioteca de Alejandría, es el sistema de control de voz de Amazon. Como se ha mencionado anteriormente, existen varios dispositivos como altavoces inteligentes o el altavoz Amazon Echo, Echo Plus etc.

El reconocimiento automático de voz o reconocimiento automático del habla (RAH) es una tecnología que convierte las palabras habladas en texto. El reconocimiento automático de voz es una disciplina de la inteligencia artificial, que es el primer paso para permitir que las tecnologías de voz como Amazon Alexa respondan cuando preguntamos "Alexa, ¿Qué tiempo hace hoy?".

Antes de RAH, nuestra conversación era simplemente una grabación de audio y picos en un ordenador. Con RAH, los ordenadores pueden los patrones relacionando los sonidos en un idioma determinado. Hoy en día los servicios de voz pueden entender la manera en que habla cada uno. En ciertos idiomas pueden distinguir entre diferentes acentos y responder adecuadamente.

Los usos y aplicaciones son varios. Los más comunes son:

- Control por comandos.
- Telefonía.
- Smartphones asistentes virtuales.
- Sistemas diseñados para dictado automático.

Amazon Alexa y los PLCs

Aunque la comunidad basada en Alexa es bastante extendida. No existe una solución comercial que conecte Alexa con un PLC industrial. Actualmente existen solo prototipos e intentos de usuarios individuales.

2.4 Opciones y Novedades

Raspberry Pi Industrial

Actualmente existen varias opciones para utilizar la Raspberry Pi en un entorno industrial. Una de estas opciones es el Raspberry Pi Compute module 3. El Módulo contiene la base de una Raspberry Pi 3 (el procesador BCM2837 y 1GB de RAM), así como un dispositivo flash eMMC de 4GB (que es el equivalente de la tarjeta SD en la Pi). El módulo, que parece una barra de RAM del portátil, se limita a lo esencial y no tiene interfaces externas. Esto permite a varios fabricantes utilizar la base con una periferia robusta y compatible con la industria.

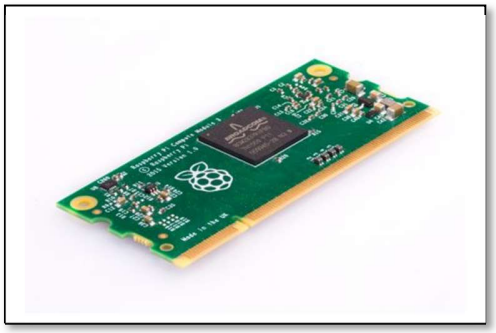


Figura 3: Raspberry Pi Compute Module 3.

Existen varias empresas y fabricantes que utilizan esta opción para sus productos finales. Además, existe la opción de utilizar Codesys que convierte una Raspberry Pi a un entorno bastante conocido en la industria. Utilizando Codesys es posible programar en “Ladder” o en “ST”. Además, de este modo, también se incorporan varios protocolos de comunicación como Profinet, Ethercat, Can etc.

Aunque lo ideal sería utilizar esta versión industrial de la Raspberry Pi en el proyecto, para disminuir el coste de un primer prototipo se ha utilizado finalmente una versión estándar de la Raspberry Pi. Como desarrollo futuro del proyecto sería interesante considerar la migración a la versión industrial de este procesador.

3. Desarrollo Proyecto.

3.1 Esquema y Arquitectura del proyecto

En la figura 4 se muestra la arquitectura del proyecto tanto de elementos físicos, protocolos de comunicación, señales eléctricas para el PLC y finalmente bibliotecas que se utilizan. Raspberry es el puente que une todos los elementos entre ellos.

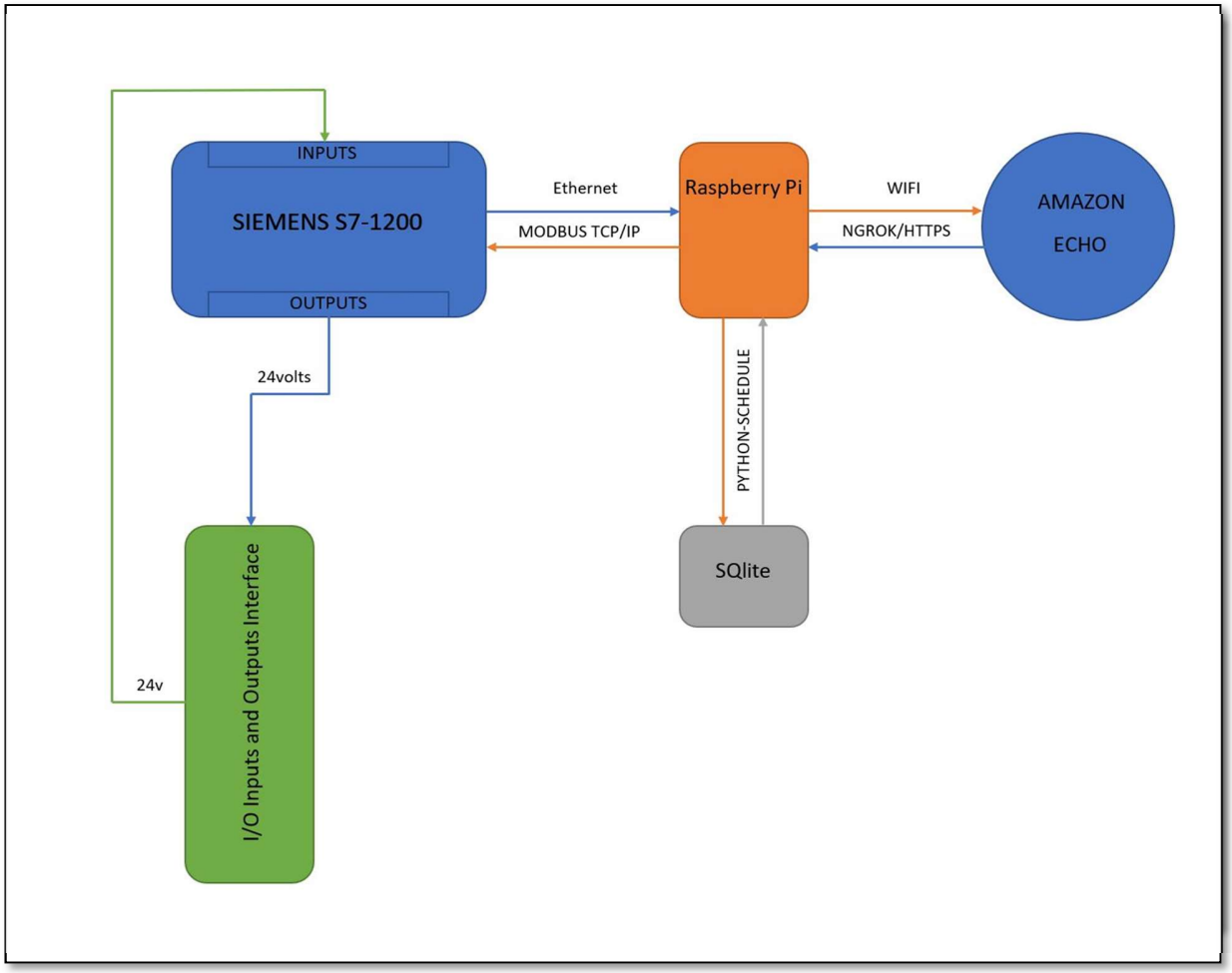


Figura 4: Arquitectura de proyecto.

- Siemens S7-1200 e Interfaz de usuario I/O: Comunicación con señales de 24v.
- Siemens S7-1200 y Raspberry Pi: Comunicación con el protocolo de comunicación Modbus TCP/IP y cable de Ethernet.
- Raspberry Pi y Amazon Echo: Comunicación con Wi-Fi vía HTTPS y Ngrok.
- Raspberry Pi y SQlite: Comunicación con él PLC vía Modbus y el Python “script” Schedule.

El montaje y conexionado de los elementos aquí descritos se describe en el capítulo 4, y se muestra en los planos que hay en el Anexo 1 del capítulo 6.

3.2 Raspberry Pi

Este capítulo se centra en describir el desarrollo de la solución propuesta en este Trabajo de Fin de Máster para usar un sistema de RAH con equipamiento industrial.

3.2.1 Instalación de Bibliotecas en la Raspberry Pi

Antes de instalar las bibliotecas es muy importante instalar PiP y actualizarlo. PiP es un sistema de gestión de paquetes que se utiliza para instalar y administrar paquetes de software escritos en Python. Se pueden encontrar en Python Package Index, para paquetes y sus dependencias.

En este proyecto todo el código está escrito con Python3. Por lo tanto, se utilizará PiP3, que descarga e instala los paquetes para Python3.

Para instalar PiP3 se utiliza el Terminal o “command-line” del sistema Rasbian. En la Figura 5 se puede ver el aspecto terminal donde se pueden introducir los comandos para instalar los paquetes pon pip3.

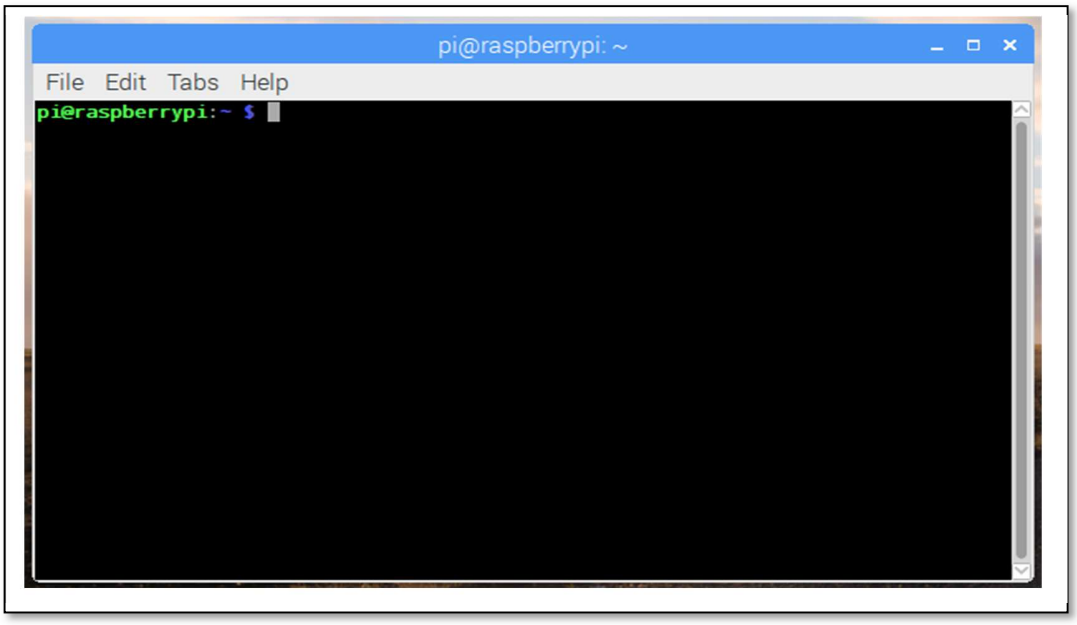


Figura 5: Command-Line.

- **PiP3**: El comando que se utiliza para que se instale es:
  pip3 install – upgrade pip3.

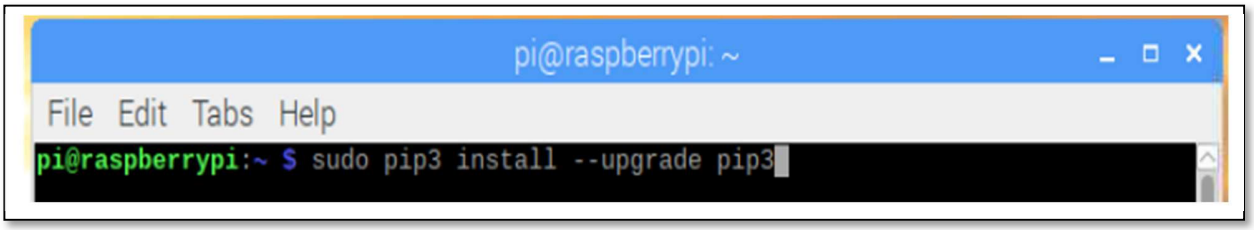


Figura 6: Command line con un comando.

- **Pymodbus**: El comando que se utiliza para que se instale es:
  PiP3 install Pymodbus

  Se trata de la biblioteca que gestiona Modbus TCP/IP

- **Ngrok**: Es una aplicación que se instala de forma independiente:
  Primero se descarga a través de la página https://ngrok.com/download.
  Una vez descargado el archivo se descomprime en el directorio Home. El directorio es /Home/pi/Ngrok-stable-linux-arm.zip

  Flask-Ask: El comando que se utiliza para instalar Flask-Ask es:
  Pip3 install flask-ask

  Sqlite: El comando que se utiliza para instalar Sqlite:
  Pip3 install sqlite3

  Python-Schedule: El comando que se utiliza para instalar Python-Schedule:
  Pip3 install schedule

3.2.2 Programación en la Raspberry Pi

Una vez acabada la fase de instalación de bibliotecas empieza la fase de programar los "scripts" que contienen todas la Bibliotecas. Según la arquitectura del proyecto es necesario comunicar con el PLC y Alexa con la Raspberry Pi. Al mismo tiempo rellenar la base de datos SQlite en intervalos. Para ello es necesario utilizar dos Scripts de Python que funcionan independientemente uno del otro.

Se diferencia en dos scripts:

- Script 1: Se gestiona la comunicación entre el PLC y la Raspberry utilizando ModBus TCP/IP. Mediante Ngrok gestiona la comunicación entre Alexa y la Raspberry Pi creando un túnel seguro. Utilizando Flask-Ask se gestionan las peticiones de Alexa a la Raspberry Pi y se mantiene abierta la sesión. Flask-Ask utiliza la base de datos SQlite bajo las peticiones de Alexa, y devuelve la información que el operario ha pedido para gestionarla con la plataforma de Amazon Alexa.
- Script 2: Crea la Base de datos si no existe, y rellena la base de datos comunicando con el PLC en intervalos de horas, días semanas etc.

Script 1

En "script" 1 se importan las siguientes bibliotecas:

- flask
- flask_ask
- pymodbus
- sqlite3

Con estas bibliotecas se puede empezar a programar el script.

A continuación, se muestra un ejemplo de la forma que se comunica Alexa con el PLC.

```
@ask.intent('ProductionYear')
def ProductionYear():
    data_bit = 0
    with ModbusClient(host='192.168.2.2',port=502) as client:
            data = client.read_holding_registers(103)
            data_bit = (data.registers[0])
    except ConnectionException as e:
            print(e)
    return question ("the total production this year is{}boxes".format(data_bit))
```

En el ejemplo anterior:

1. Se crea el objeto Ask con la aplicación Flask y una ruta para reenviar las solicitudes de Alexa.
2. El método "intent" prepara la función "ProductionYear" para que sea visible y accesible
3. "ModbusClient" utiliza la dirección IP y el puerto para comunicar con el PLC.
4. "Client.read_holding_register()" lee la zona de memoria en el PLC.
5. "Data.register[0]" se utiliza para para pasar los datos a una variable.

6. “ConnectionException” gestiona los errores de conexiones.
7. Devuelve la información que ha solicitado Alexa.

Script 2

En “script” 2 se importan las siguientes bibliotecas:

- schedule
- sqlite
- Pymodbus

A continuación, se muestra un ejemplo de la forma que se rellena la base de datos con el PLC.

```
def daily_production():
    data_int = 0
    with ModbusClient(host='192.168.2.2',port=502) as client:
         data = client.read_holding_registers(100)
         data_int = data.registers[0]
    except ConnectionException as error:
         print(error)
    conn = sqlite3.connect("plc_data.db")
    c = conn.cursor()
    c.execute("INSERT INTO Daily_Production (Production)VALUES(?)",(data_int,))
    conn.commit()
    c.close()
```

En el ejemplo anterior:

1. Se crea la función "daily_production". El módulo Phyton-Schedule llamará a esta función cada día.
2. “ModbusClient” utiliza la dirección IP y el puerto para comunicar con el PLC.
3. “Client.read_holding_register()” lee la zona de memoria en el PLC.
4. “Client.read_holding_register()” lee la zona de memoria en el PLC.
5. “Data.register[0]” se utiliza para para pasar los datos a una variable.
6. “ConnectionException” gestiona los errores de conexiones.
7. “sqlite3.connect()” se conecta con la base de datos.
8. “c.execute()” rellena la base de datos que esta almacenado en la variable de “Data.register[0]”.
9. “c.close” cierra la conexión con la base de datos.

Com se ha comentado anteriormente, el “script” 2 crea la base de datos si no existe anteriormente. Dentro de la base de datos se puede almacenar información de tipos Bool,

Integer, Null, Real, Text o Blob, aunque Modbus es el factor que limita la información que se puede intercambiar a los tipos Integer (para valores) y Bool (para bits).

Primero se crean las tablas que se van a utilizar, como por ejemplo “Daily_Production”. En cada tabla se crea un identificador ("id") para cada dato y finalmente las columnas con la información que se va a rellenar la base de datos

Se crean 3 tablas diferentes en la base de datos.

1. **Tabla** “Daily_Production”:
   Dentro de esta tabla se crean dos columnas.
   Columna “Id”-> Identificador de los datos que se almacenan dentro de “Daily_Production”.
   Columna “Production”-> Los datos que se almacenan diariamente en la base de datos.
2. **Tabla** “Weekly_Production”:
   Dentro de esta tabla se crean dos columnas.
   Columna “Id”-> Identificador de los datos que se almacenan dentro de “Weekly_Production”.
   Columna “Week_Production”-> Los datos de producción que se almacenan cada semana en la base de datos.
3. **Tabla** “Yearly_Production”:
   Dentro de esta tabla se crean dos columnas.
   Columna “Id”-> Identificador de los datos que se almacenan dentro de “Yearly_Production”.
   Columna “Year_Production”-> Los datos de producción que se almacenan cada año en la base de datos.

3.3 S7-1200 Siemens PLC

3.3.1 Nuevo proyecto y ajustes de PLC

Siemens S7-1200 se programa con el TIA Portal, como se describe en el apartado 2.2.2. En este punto la filosofía de programación cambia radicalmente en comparación con Raspberry y los "scripts" en Python. TIA Portal tiene incorporada toda la información y funciones que se necesita para programar el PLC. No existen bibliotecas externas o soluciones fuera de lo que ofrece Siemens con TIA Portal. La versión que se utiliza en el proyecto es TIA Portal 15, es la última versión que existe actualmente.

TIA Portal separa la programación en varias zonas, una de ellas es la zona de programación de PLCs. También se puede programar dispositivos para HMI (Human-Machine Interface), objetos tecnológicos, variadores de frecuencia, etc. En el proyecto, se selecciona y programa solo el apartado de PLCs.

Una vez que TIA portal ha arrancado, y después de seleccionar la programación de PLCs, hay que añadir un nuevo dispositivo con la opción:

- Project.
  - New.
    - Add New device.

En la nueva ventana que aparece se selecciona el PLC que se utiliza. Es este proyecto se utiliza el modelo S7-1200 1214 DC/DC/DC. Después de seleccionar el PLC, éste aparece en la lista "Devices&Networks", como muestra la figura 7.

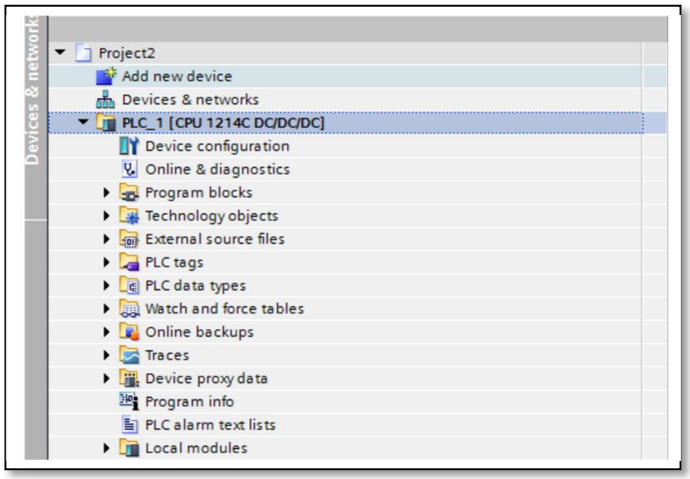


Figura 7: Nuevo proyecto TIA Portal con PLC seleccionado.

Una vez definido el tipo de PLC que va a utilizar, aparece la opción “Device Configuration”. Dentro de “Device Configuration” se utilizan las siguientes opciones, como muestra la figura 8:

- **General**: Se almacena la información de nombre que tiene el proyecto. También se almacena el nombre de autor y comentarios sobre el proyecto.
- **PROFINET interface [X1]**: En ***PROFINET*** se define el nombre de PLC que se utiliza dentro de la red. En “E*thernet Addresses”* se añade un nuevo “subnet PN/IE_1” para la red de ModBus. En **IP-Protocol** se define la dirección IP y la submascara del PLC.
- **System and clock:** En *“System memory bits”* se activa la opción “*enable the use of system memory byte”.* Con esta opción activada se puede utilizar bits bits como los que indican el "primer ciclo de scan", "siempre activado", etc. También se activa la opción de “Clock memory bits”. Esta opción activa bits que se activan y desactivan automáticamente con varios intervalos.
- **Protection & Security:** Se desactiva la contraseña y se activa el “Full Access”.

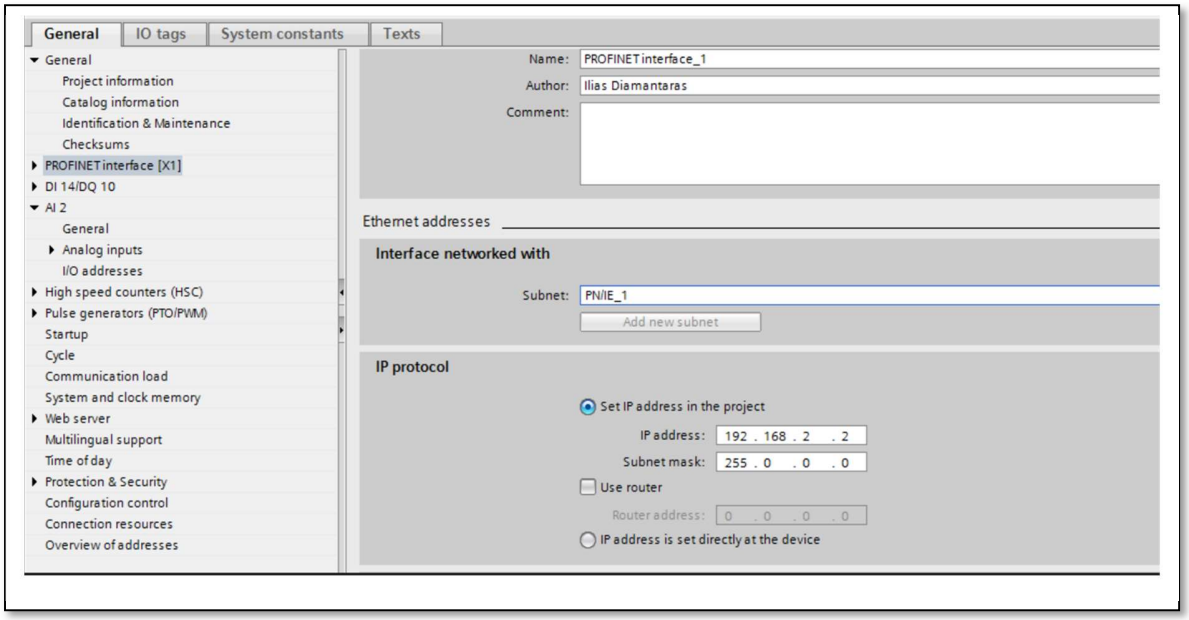


Figura 8: Ajustes de PLC.

Por otro lado, es posible ajustar las entradas y salidas de PLC DI 14/DQ 10. Se puede ajustar que tipo de detección es la entrada, flanco ascendente o flanco descendente. La salida se puede controlar si mantiene o no el ultimo valor en el caso que se apaga el PLC.

Es importante notar que existen muchas más opciones que se pueden utilizar, pero no son necesarias para este proyecto.

3.3.2 Programación de PLC

Una vez acabada la fase de configuración, se puede programar el PLC. Antes de empezar a programar el PLC es necesario definir las variables. Las variables en los PLCs generalmente se separan en las globales "Global" o locales "Local". Las variables que se declaran en PLC *"***Tags**" o en un "**Program Block**" forman parte de las variables globales y pueden ser accesibles desde todos los bloques de programa. Las variables y parámetros declarados en la zona de declaración de parámetros del bloque son operaciones locales que solo se pueden utilizar en el mismo bloque declarado.

Las tablas de variables del PLC contienen la declaración (definición) de las variables y los valores válidos para las variables globales y las constantes del usuario. Cuando se añade el PLC se crea automáticamente una tabla de variables. La tabla de variables del PLC contiene una pestaña para variables, otra para constantes de usuario y otra como constantes del sistema.

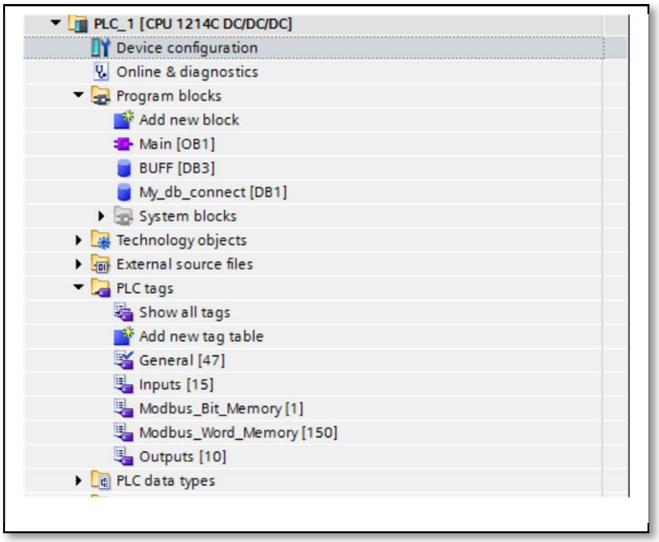
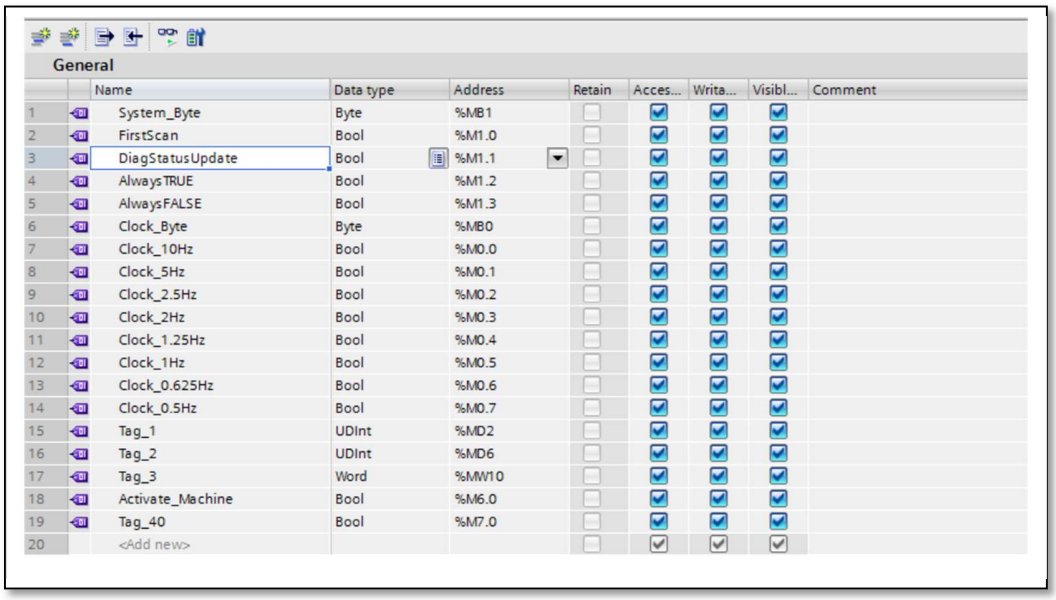


Figura 9: Árbol de PLC.

Las tablas de variables se pueden crear haciendo "click" en la opción "*Add new tag table"*. En este momento el PLC crea una nueva lista que se puede rellenar con varias variables con diferentes nombres, tipo de variable y dirección. Todas las variables globales, como entradas, salidas o marcas pueden direccionarse por simbólico o con direccionamiento absoluto.

El PLC mantiene el valor de las memorias una vez apagado el sistema. Esto se llama memoria remanente y cada PLC tiene la opción que declarar las variables que son remanentes y las que no. La utilización de las memorias remanentes del PLC configurada se muestra "*offline"* en el árbol de proyecto.



General

	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...	Comment
1	System_Byte	Byte	%MB1		☑	☑	☑	
2	FirstScan	Bool	%M1.0		☑	☑	☑	
3	DiagStatusUpdate	Bool	%M1.1		☑	☑	☑	
4	AlwaysTRUE	Bool	%M1.2		☑	☑	☑	
5	AlwaysFALSE	Bool	%M1.3		☑	☑	☑	
6	Clock_Byte	Byte	%MB0		☑	☑	☑	
7	Clock_10Hz	Bool	%M0.0		☑	☑	☑	
8	Clock_5Hz	Bool	%M0.1		☑	☑	☑	
9	Clock_2.5Hz	Bool	%M0.2		☑	☑	☑	
10	Clock_2Hz	Bool	%M0.3		☑	☑	☑	
11	Clock_1.25Hz	Bool	%M0.4		☑	☑	☑	
12	Clock_1Hz	Bool	%M0.5		☑	☑	☑	
13	Clock_0.625Hz	Bool	%M0.6		☑	☑	☑	
14	Clock_0.5Hz	Bool	%M0.7		☑	☑	☑	
15	Tag_1	UDInt	%MD2		☑	☑	☑	
16	Tag_2	UDInt	%MD6		☑	☑	☑	
17	Tag_3	Word	%MW10		☑	☑	☑	
18	Activate_Machine	Bool	%M6.0		☑	☑	☑	
19	Tag_40	Bool	%M7.0		☑	☑	☑	
20	<Add new>				☑	☑	☑	

Figura 10: Variables de PLC.

Una vez definidas las variables del PLC el siguiente paso es crear los bloques. El TIA portal dispone de varios tipos de bloques en los que se almacenen los datos y las instrucciones de programa. Dependiendo de los requisitos del proceso, el programa entero puede estructurarse en diferentes bloques. Se permite el uso de todas las instrucciones en todos los bloques (FB, FC y OB). Para crear un nuevo bloque se hace “click” en “*Add new block”* ver figura 9.

Los bloques de organización o OB “Organization block” forman parte del interfaz de usuario frente al interfaz de sistema. El programa entero puede almacenarse en OB1 ya que se ejecuta cíclicamente, o se puede dividir en varios bloques.

Es importante notar que existen varios tipos de bloques de organización. El principal es OB1, pero es posible utilizar otro tipo de bloques. Permiten ejecutar el código que se contiene dentro de estos bloques en especificas condiciones. Un ejemplo es OB100 que solo se ejecuta el código al inicio del sistema, es decir el primer ciclo de scan. Se puede asignar el código según necesidades de cada aplicación, algo que facilita la programación de cada proyecto.

Las funciones o FCs “Function” contienen las funcionalidades parciales de un programa. Cabe la posibilidad de programar funciones con parámetros que serán asignados en la llamada de la función. Como característica, las funciones también son adecuadas para programas reiterativos, con funcionalidades complejas.

Los bloques de función o FBs “Function blocks” ofrecen las mismas posibilidades que las funciones. Además, los bloques de funciones se vinculan con un área de memoria de un bloque de datos de instancia. Como característica, los bloques de función son adecuados para programas reiterativos con funcionalidades complejas.

En este proyecto se utiliza la programación lineal. Considerando el pequeño tamaño del programa, no se requiere otro tipo de programación como puede ser una dividida por tareas o estructurada. En programación lineal, el programa entero se puede escribir en un mismo bloque. Este modelo se asemeja a un control de reles cableados que fuera reemplazado por un sistema de automatización actual (PLC). La CPU procesa las instrucciones individuales una tras las otra.

La CPU trata la ejecución del programa cíclicamente, por lo que todo el código se ejecuta en un bucle constante. Después de ejecutar el ciclo de programa, vuelve a ejecutarse de nuevo. En cada ciclo la CPU ejecuta los siguientes pasos:

- La CPU consulta los estados de las señales de entada y actualiza los valores en la imagen de proceso de entradas.
- Secuencialmente, la CPU procesa las instrucciones programadas y trabaja directamente las imágenes de proceso y no sobres las E/S.
- La CPU transfiere los estados de salida desde la imagen de salidas a los módulos digitales de salidas.
- Tiempo de ciclo y observación de ciclo.
- El tiempo que la CPU necesita para la ejecución completa de un ciclo de programa, es el tiempo de ejecución que requiere el sistema operativo de la CPU para su control. Si el tiempo de vigilancia excede el más del doble tiempo de ejecución, la CPU pasara a stop.

El Código de programa en el OB1 es ejecutado cíclicamente. Con el programa cíclico de ejecución, el tiempo de ejecución o ciclo de scan depende del sistema operativo de la CPU y de la suma de todos los comandos ejecutados.

Como se ha comentado anteriormente, TIA Portal tiene incorporados todos los bloques de tareas comunes en programas de automatización. Una de ellas que se hace uso en este proyecto es “MB_SERVER”, que habilita el PLC como servidor Modbus-TCP a través de una conexión Profinet. La instrucción procesa las solicitudes de conexión de un cliente Modbus TCP, recibe y procesa y envía respuestas.

Se puede añadir la instrucción del apartado de Intrusctions→Communication→MODBUS TCP al programa, arrastrando el bloque, de forma que esta instrucción gestione la comunicación con la Raspberry Pi.

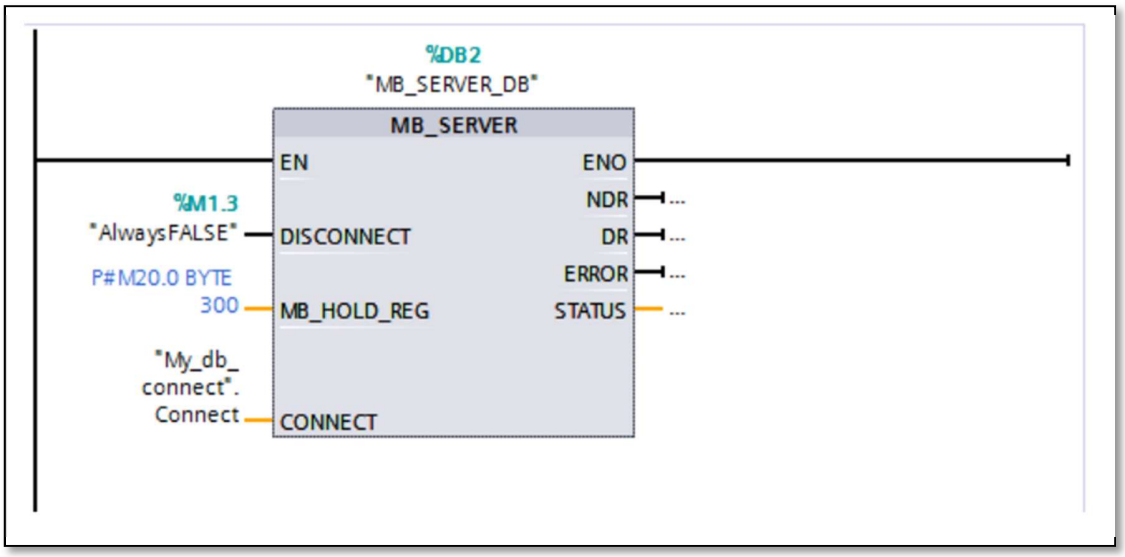


Figura 11: Instrucción MB_Server.

Para que la instrucción MB_SERVER funcione correctamente, se tiene que añadir las variables en los parámetros de entrada, que son los siguientes:

- **DISCONNECT:** Se puede utilizar este parámetro para controlar cuando las peticiones de conexión serán aceptadas o no. Si es 0 se establece una conexión pasiva, de esta manera Raspberry Pi siempre puede comunicar con el PLC. Si es 1 se inicializa la terminación de conexión. Se ha utilizado la variable "Always False" que siempre es 0.
- **MB_HOLD_REG:** Puntero de Modbus para la memoria dentro de PLC. Siempre se referencia a una zona de memoria más grande que dos bytes. Utilizando la variable P#M20.0 BYTE 300 se establece dónde empieza y termina la zona de memoria que utiliza la instrucción.
- **CONNECT:** Puntero de la estructura de la descripción de la conexión. Según el manual de S7-1200 los parámetros que se utilizan, se pueden ver en la figura 12.

Name	Data type	Start value
Static		
Connect	TCON_IP_v4	
InterfaceId	HW_ANY	64
ID	CONN_OUC	1
ConnectionType	Byte	11
ActiveEstablished	Bool	false
RemoteAddress	IP_V4	
RemotePort	UInt	0
LocalPort	UInt	502

Figura 12: MB_SERVER CONNECT parámetros.

En el caso de las salidas que tiene la instrucción:

- **NDR:** la salida se establece a 0 si no tiene nuevos datos y 1 si el cliente Modbus está escribiendo nuevos datos.
- **DR:** la salida se establece a 0 si no está leyendo datos y 1 si el cliente está leyendo.
- **Error:** Si se produce un error durante la llamada del servidor MB_SERVER, la salida del parámetro de error se establece en 1.
- **STATUS:** Información detallada de la instrucción.

Utilizando programación Ladder se puede leer el tipo de error. En el momento que se activa la salida error la instrucción "MOVE", pasa el error a una palabra "Tag_3". Dentro de esta palabra se almacena un código. Utilizando el manual se puede consultar el tipo de error y posibles soluciones.

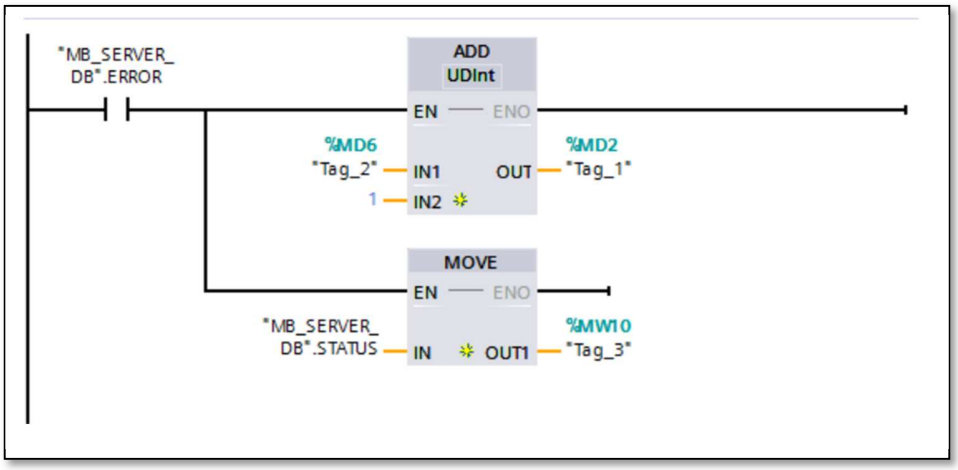


Figura 13: "MB_SERVER" Error STATUS.

Una vez programado el "MB_SERVER" se puede comunicar con la Raspberry Pi. El siguiente paso es definir las variables que intercambia el PLC con la Raspberry PI. Para ello se crea en el árbol de las variables dos distintos grupos.

El primer grupo es "Modbus_Bit_Memory", aquí se asignan todas las variables tipo Bit que utiliza el PLC para la comunicación. El segundo grupo es "Modbus_Word_Memory", aquí se asignan todas las variables tipo Word. De esta forma es fácil acceder la memoria según la necesidad del programa. En la figura 9 se puede observar los grupos de memoria.

Por otro lado, se crean dos grupos llamados "Inputs" y "Outputs". Se asignan las entradas y salidas digitales de PLC respectivamente. Con Modbus es posible activar las salidas directamente, también es posible leer el estado actual de las entradas.

Toda la secuencia de programa está hecha según las necesidades del proyecto. Como se ha comentado anteriormente el PLC simula una máquina, el programa es una simulación de cómo se intercambia o interactúa con los datos y con la Raspberry Pi.

3.4 Programación Amazon Alexa

3.4.1 Programación del Amazon Alexa

Amazon Alexa es la plataforma donde se puede programar las interacciones con el usuario. En la figura 14 se muestra el diagrama de flujo que se utiliza, así como la arquitectura de procesamiento de información, y como se trata dentro de la plataforma de Alexa y Raspberry Pi.

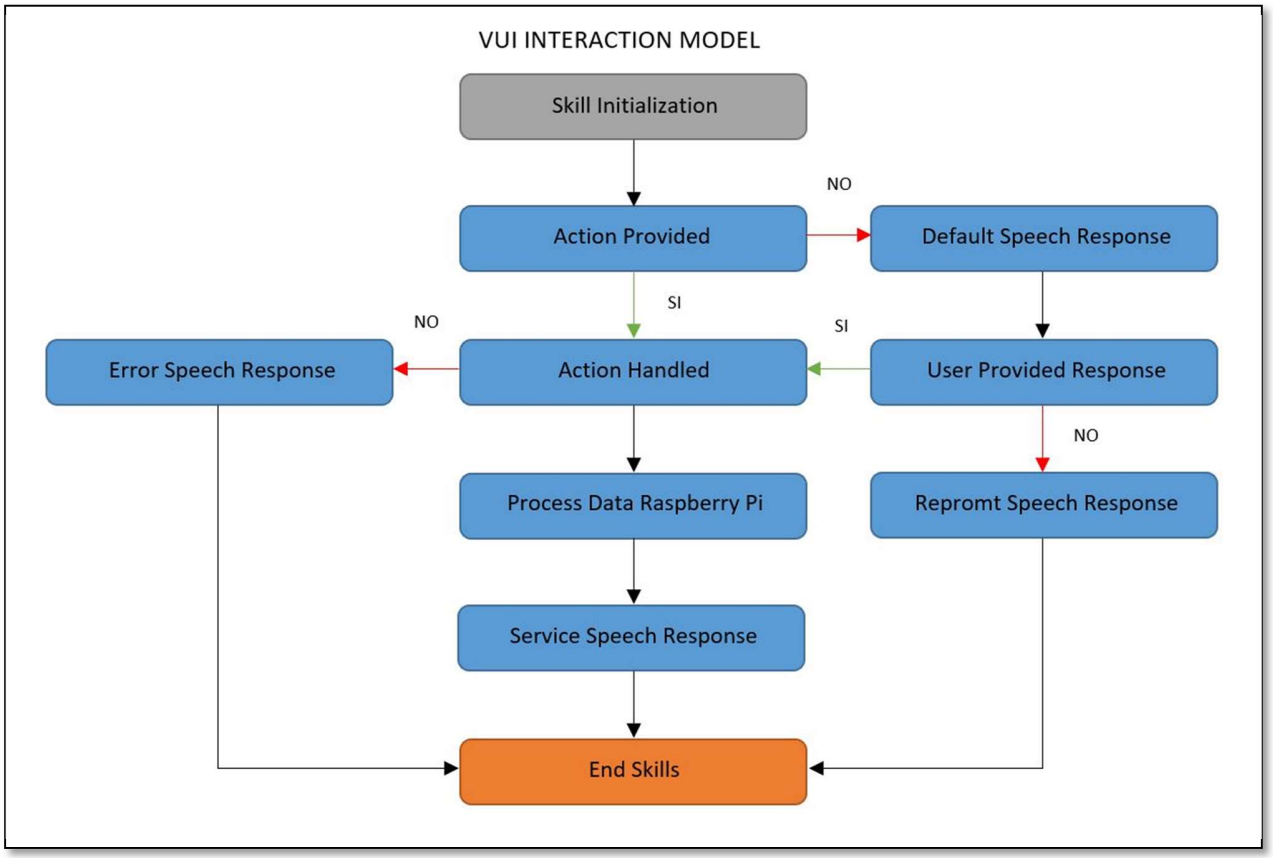


Figura 14: Arquitectura de VUI.

Alexa es el servicio de voz ubicado en la nube de Amazon, y disponible en los dispositivos de Amazon, así como en otros dispositivos de terceros con Alexa integrada. Además, cuenta con unas funcionalidades que Amazon llama “skills”, las cuales permiten crear una experiencia más personalizada.

El Alexa Skills Kit (ASK) es un conjunto de herramientas, documentación, muestras de código y API con el que se puede añadir “skills” a Alexa de forma rápida y sencilla. El ASK permite a diseñadores y desarrolladores crear "skills" atractivos para los consumidores finales.

Con este kit, se puede aprovechar el conocimiento y la innovación de Amazon en el sector del diseño de voz.

Para desarrollar un “skill” de Alexa utilizando la API de Amazon, es necesario crear una cuenta de desarrollador en "Amazon Developer Services". En la plataforma de Alexa se diseña el modelo de interacción de voz en la Consola de desarrollador de ASK y luego se conecta a un servicio final que se puede implementar en AWS Lambda, un servicio de informática sin servidor proporcionado por Amazon, o en un servidor HTTPS, como puede ser la Raspberry Pi.

Una vez que ya se dispone de una cuenta de desarrollador de Amazon, se puede seguir los siguientes pasos para crear un "skill" de Alexa:

1. Acceder al portal de desarrolladores de Amazon. En la esquina superior derecha de la pantalla existe el botón “**Sign In**”.
2. Una vez que se ha iniciado sesión, en la esquina superior derecha se selecciona el enlace de “**Create Skill**”.
3. Se asigna un nombre para el "skill" que se va a desarrollar. En este caso se ha llamado **PLC**
4. Se selecciona el botón “**Custom**” para agregarlo al “skill”.
5. Finalmente se selecciona el botón “**Create skill**”.

La plataforma que aparece en la figura 15, es la plataforma de Amazon Alexa que se utiliza para crear “skills”. Aquí se puede desarrollar y construir el modelo de interacción con el usuario final.

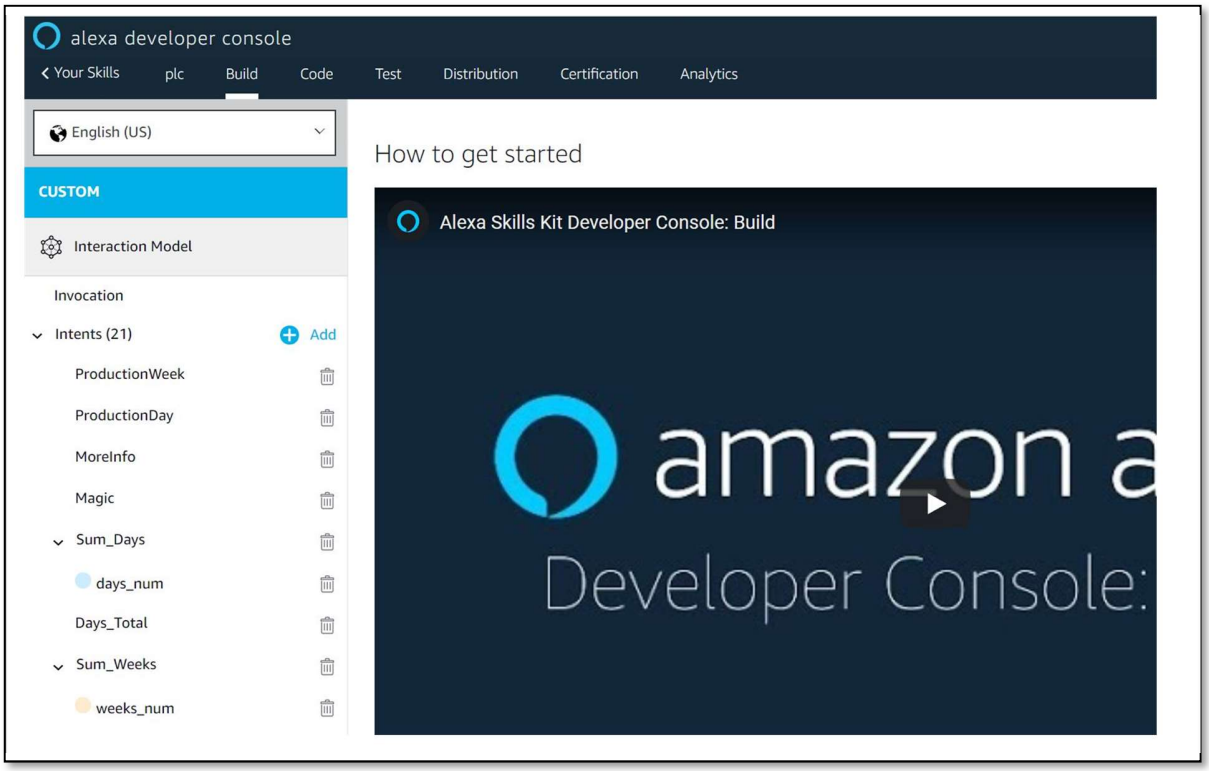


Figura 15: Plataforma de Amazon Alexa Developer.

Dentro de la plataforma de Alexa, en el panel de la izquierda se muestra un árbol con varias opciones para desarrollar el "skill". El primer paso es asignar el nombre de invocar el “skill”. Los usuarios dicen el nombre de la invocación para comenzar la interacción con el “skill” desarrollado. Por ejemplo, si el nombre de invocación para el “skill” **PLC** es "**My Machine**", los usuarios pueden decir “Echo activate my machine”. Existen varias maneras de activar el “skill”, como “open”, “launch” etc. El nombre que se ha seleccionado es “my machine”, es importante tener en cuenta que no se pueden utilizar letra mayúscula.

Una vez asignado el nombre con el que se puede invocar el “skill” **PLC**, se añaden los “**intents**”. Un “intent” representa una acción que cumple con la solicitud hablada de un usuario. Los “intents” pueden tener opcionalmente argumentos llamados “slots”. Existen "intents" estándar ya definidos por Amazon, que se utilizan para acciones comunes y generales, como detener, cancelar y pedir ayuda. Mientras que en los "intets" definidos por el usuario se puede añadir o cambiar los "slots", en la mayoría de los "intents" predefinidos por Amazon no es posible. Algunos ejemplos son “Amazon.CancelIntent” que cancela la acción actual o “Amazon.HelpIntent” que proporciona ayuda sobre cómo usar el “skill”.

Figura 16: Alexa “intents”.

Igual que el “skill” tiene su nombre de invocación y el usuario puede llamar a este "skill" en cualquier momento, cada “intent” tiene su nombre llamado “**Sample Utterance**”. Una vez que el usuario a invocado el “skill” PLC, puede proporcionar una expresión o “utterance” para que Alexa tenga suficiente información y activar el “intent” que el usuario está llamando. La facilidad de uso del Skill depende directamente de cómo los “utterance” de muestra y los valores de los “slots” personalizados representan el uso del lenguaje en el mundo real. Cada "skill" puede tener un "utterance" o varios, siendo lo segundo más común, y así el usuario final no se limita a sola una expresión, sino a varias más, como en una conversación con otra persona.

Cada “intent” tiene la capacidad de opcionalmente asignar diferentes variables, llamadas “slots”. Los “slots” son básicamente variables en expresiones. Estos pueden tener valores predefinidos, pero por defecto están vacíos. Para definir un “slot”, primero se debe crear un “intent” personalizado. Después de crearlo, cuando es necesario crear expresiones de muestra para un "intent", basta con definir un "slot" escribiéndolo entre llaves.

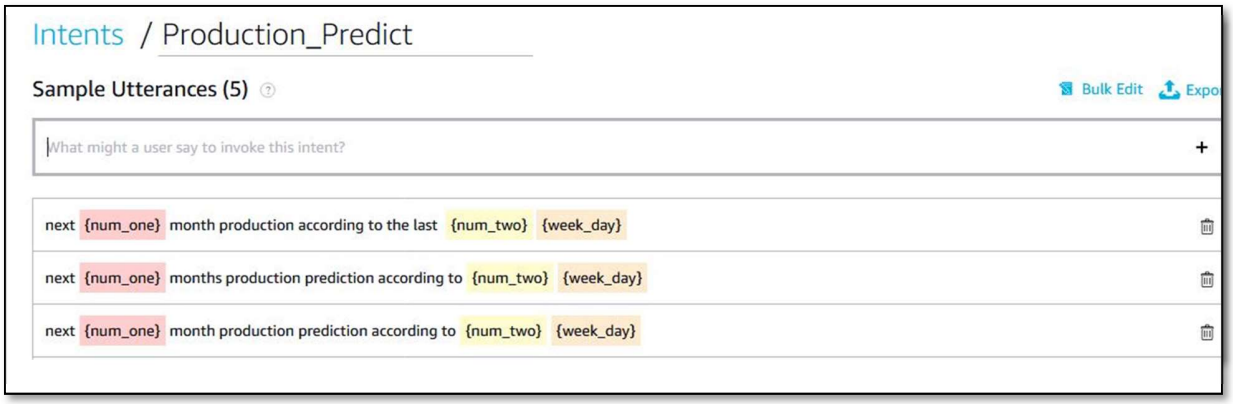


Figura 17: Sample Utteraces y "slots".

En la figura 17 se muestra un “intent” llamado “Production_Predict”. La plataforma detecta automáticamente los “slots” entre llaves y lo resalta. En la lista Sample Utterances se puede observar observar 3 tipos distintos de "slots". El primero **{num_one}** y el segundo **{num_two}** utilizan un tipo de “slot” predefindo por Amazon. El tipo que utilizan es “Amazon.Number”, se utiliza para convertir números o palabras que expresan un número en dígitos, incluidos números decimales. El “slot” **{week_day}** es un “slot” creado para este “intent” en particular.

En la parte inferior del árbol de la plataforma Alexa, existe la opción “**Slot Types**”, como se muestra en la Figura 17. Dentro de esta opción, seleccionando "add", se crea un nuevo "slot". Una vez creado el “slot” se asigna el nombre week_day”. De este modo, se puede añadir nuevos que Alexa selecciona cada vez que se utiliza el “slot” en un “intent”. Los valores que se han añadido son “Day”, “Days”, “Week” y “Weeks”.

Cada vez que el usuario quiere llamar el Intent “Production_Predict” utiliza las expresiones en Sample Utterances. Alexa procesa toda la frase y pronuncia los "slots" {num_one} y {num_two} a formato numérico. El "slot" {week_day} pasa a formato de texto los valores que tiene asignados: “Day”, “Days”, “Week” y “Weeks”.

Es posible editar y asignar nombres directamente en el editor JSON que ofrece la plataforma de Amazon Alexa. JSON es el acrónimo de “Java Script Object Notation”. JSON es un formato para almacenar la información de una manera fácil y organizada. Es un estándar basado en texto plano

para el intercambio de información, por lo que se usa en muchos sistemas que requieren mostrar o enviar información para ser interpretada por otros.

Es posible añadir hasta 250 intents en cada “skill”. Cada “skill” es posible tener un total de 50.000 valores de “slots” personalizados. Los “sample utterances” pueden llegar hasta 200.000 para cada “skill”. Esto permite desarrollar cualquier tipo de Interfaz de Voz.

Todos los "intents" desarrollados tienen que ser gestionados por la Raspberry Pi con la librería de Flask-Ask. La comunicación entre la Raspberry Pi y Alexa se hace a través de HTTPS. Con el Kit de “skills” de Alexa, las frases habladas se asignan a acciones ejecutadas en un servidor, y convierte el habla en un archivo JSON y entrega ese archivo a su aplicación. Por ejemplo, la frase: "Alexa, give me the production for today”, se gestiona en la plataforma de Alexa y se entrega en formato de texto en la Raspberry Pi. En la parte izquierda abajo en el árbol se selecciona “endpoint”. El “endpoint”, que en este caso es la Raspberry Pi, recibirá solicitudes POST cuando un usuario interactúe con el “skill” de Alexa. El cuerpo de la solicitud contiene parámetros que el servicio puede usar y realizar la lógica y generar una respuesta con formato JSON.

Los “endpoints” son aspectos importantes de la interacción con las API web del lado del servidor, ya que especifican dónde se encuentran los recursos a los que puede acceder un software de terceros como la Raspberry Pi. En la figura 18 se muestra la introducción del “endpoint” en la plataforma de Amazon Alexa. Por lo general, el acceso se realiza a través de un URI, en este proyecto la genera Ngrok “https://34dda1d8.ngrok.io/”, al que se publican las solicitudes HTTPS y, por lo tanto, se espera la respuesta. Los “endpoints” deben ser estáticos, de lo contrario no se puede garantizar el correcto funcionamiento del software que interactúa con él.

Dentro de la configuración de “endpoint” existen dos opciones, como se puede ver en la Figura 18: AWS Lamda ARN, y HTTPS. En este proyecto se emplea HTTPS. Además, dentro de las tres posibles opciones del campo "Default región" hay que seleccionar “My development endpoint is a sub-domain that has a wildcard certificate from a certificate authority”.

Figura 18: Plataforma Amazon Alexa “endpoint”.

Por último, con la opción “invocation”, que está en la parte superior del árbol del panel izquierdo del entorno de Alexa (ver Figura 15), se puede guardar y construir el proyecto. Además, con la opción "Skill builder checklist", que está en el panel de la derecha del entorno, puede ver la lista de verificación, que sirve para comprobar si el "skill" está construido correctamente.

3.4.2 Amazon Echo

Para este proyecto, el altavoz que se ha utilizado es el Amazon Echo de segunda generación. La segunda versión de Amazon Echo solo permitía inglés, alemán y japonés. Se ha actualizado la segunda generación con la opción de utilizar español y varios idiomas más.

Figura 19: Altavoz Amazon Echo.

Antes de utilizar el altavoz hay que registrarlo en la página web "http://alexa.amazon.com". Los pasos para registrar el Echo son:

1. Se conecta el Amazon Echo mediante Wi-Fi.
2. Dentro de la página de Amazon Alexa se selecciona Settings.
3. La primera opción es "Set up new device". Se selecciona esta opción.
4. Seguir los pasos que indica en la página para registrar el altavoz.
5. Una vez conectado se despierta utilizando la palabra “Echo”.

Utilizando la misma cuenta en Amazon Developer y Amazon Alexa se enlaza automáticamente el altavoz con el “skill” que se ha desarrollado. Utilizando la frase “Echo, activate my machine” activa el “skill” PLC que se ha desarrollado para este proyecto.

La red Wi-Fi que utiliza el altavoz Echo de Amazon, es una conexión estándar a una red con acceso a internet. Es decir, cualquier tipo de conexión Wi-Fi con acceso a internet, aunque no es la misma que utiliza la Raspbbery Pi podría valer. Es importante tener en cuenta que la conexión es necesario ser estable, para facilitar la conexión y una respuesta rápida con los servers de Amazon.

3.5 Desarrollo de Cuadro Eléctrico

3.5.1 Cuadro Eléctrico

El cuadro o armario eléctrico es un elemento clave en el control de una máquina, o cualquier tipo de instalación de automatización, y normalmente cada cuadro eléctrico está diseñado para una aplicación específica. Un cuadro contiene un controlador o varios (habitualmente PLCs), los preaccionamientos necesarios (relés, variadores de frecuencia, electrovávulas, etc.), fuentes de alimentación, y otros elementos de interfaz entre el controlador y la máquina, además del cableado necesario. Además, un cuadro eléctrico suele incluir elementos HMI (interfaz hombre-máquina), que pueden ser desde pulsadores o conmutadores simples, hasta pantallas táctiles.

En este proyecto, se ha diseñado un cuadro eléctrico que permite simular una máquina. Además, este cuadro permite al usuario interactuar con la máquina mediante el altavoz Echo, facilitando el acceso a la plataforma de Alexa.

Los materiales que se han utilizado para el cuadro eléctrico son materiales industriales. Esto permite simular una máquina que posee características similares de lo que encontraría un usuario en un entorno industrial. El cuadro eléctrico es de superficie, es decir se apoya sobre una superficie, esta superficie es de madera y facilita la instalación de los elementos eléctricos que se utilizan.

Los materiales que han utilizado son:

Canaleta Ranurada: Vienen a varios tamaños y características. Se utiliza para facilitar el montaje y ofrece mejor conducción del cableado.

Carril DIN: Un carril DIN es una barra de metal utilizada para el montaje de magnetotérmicos y equipos de control industrial dentro de los cuadros eléctricos. Generalmente están hechos de acero con un acabado de superficie brillante galvanizado o cromado.

Bornero: Los borneros se utilizan para la conexión dentro de los cuadros eléctricos. La borna es el punto de conexión entre la instalación del cuadro y la máquina. La conexión de la borna tiene dos destinos, el aparato del cuadro, y la utilización del cableado con el destino final sea una máquina o una instalación.

Cableado: Los cables que se han utilizado tienen varias secciones y colores. La entrada para la alimentación de cuadro es monofásica.

La sección de los cables para la alimentación del cuadro es de 2,5mm. Esto permite utilizar equipos hasta 15A.

La sección de los cables para la maniobra dentro del cuadro eléctrico es de 0.5mm. La tensión que se utiliza es de 24v continua. La mayoría de la sensorica y PLCs trabajan en esta tensión. Los colores que se utilizan son, rojo para el positivo y azul para el negativo. La entrada de PLC es de color violeta y la salida de color negro.

Los elementos eléctricos, de control etc. que se van a montar en el cuadro eléctrico son:

Fuente de Alimentación: La fuente es el modelo *S8VK – G06024*. El número de parte indica las características que tiene este modelo.

G = Monofásico 100-240v

060 = 60W

24 = 24V

PLC: Se emplea un PLC S7-1200 de Siemens. En la actualidad, este tipo de PLC se usa en todo tipo de aplicaciones en procesos y secuencias de maquinaria. El modelo que se utiliza es la *CPU 1214C DC/DC/DC* y sus características son:

Tensión de Carga: 24V y el rango admisible es 20,4-28,8V

Entradas digitales: 14 integradas

Salidas digitales: 10 integradas

Entradas analógicas: 2 entradas con canales integrados de 0 a 10V

Interfaz de comunicación: Puerto ethernet que utiliza el protocolo Profinet, S7o Modbus.

Ethernet Switch: El switch de Weidmuller *IE-SW-BL08-8TX* funciona como una estación central que conecta el PLC, Raspberry y cualquier otro dispositivo con un puerto ethernet. El switch también está conectado a una red con acceso a internet.

Magnetotérmico: El magnetotérmico se utiliza para la protección de los dispositivos. Es capaz de interrumpir la corriente eléctrica de un circuito cuando ésta sobrepasa ciertos valores máximos. Se utiliza el modelo C60N de Schneider, de 3A.

Panel de pulsadores e indicadores: La botonera instalada en el cuadro simula la parte física que interactúa el usuario con una máquina. Los selectores que se han instalado se conectan con las entradas digitales del PLC. Además, se incorpora señalización de LED que permite visualizar las salidas digitales del PLC.

“Encoder”: El “encoder” que se utiliza es de la marca Sick. Es un dispositivo electromecánico usado para convertir la posición angular del eje de un motor a un código digital que puede leer el PLC. El “encoder” que se utiliza es el DFS60B-S4EC00500. Cuenta hasta 500 pulsos cada revolución y trabaja con 3 fases de referencia de posición. Su tensión de alimentación es de 10V a 32V.

Raspberry Pi: Es el ordenador que gestiona la comunicación con Amazon Alexa a través de Internet, y las peticiones al PLC mediante Modbus.

Interfaz de red Ethernet con conexión USB: Convierte el puerto USB a ethernet. Se utiliza para comunicar con el PLC.

Amazon Echo: Altavoz de Amazon.

Los dispositivos y los materiales que se utilizan son indispensables para el correcto funcionamiento del proyecto y la comunicación entre los dispositivos. Como se ha mencionado anteriormente, la selección de estos elementos se ha hecho teniendo en cuenta que se tiene que simular una máquina.

3.5.2 Esquema Eléctrico

El esquema eléctrico generalmente da la información sobre la conexión entre ellos sin atender a su ubicación real. La posición específica de los dispositivos en el cuadro eléctrico se indica con planos de posición adicionales. El programa que se ha utilizado para el esquema eléctrico y los planos es AutoCAD, de la casa Autodesk.

El esquema eléctrico y los planos de posición de los elementos en el cuadro eléctrico conforman 8 páginas que se encuentran en el Anexo 1 "Esquema Eléctrico". Son las siguientes:

Página 1: Se define la alimentación del cuadro eléctrico. La tensión es 220V entre fase y neutro pasando primero por el magnetotérmico que protege la instalación. Se conectan en 220V la fuente de alimentación y el Enchufe.

Página 2: Se define la instalación de la maniobra en 24V. Se puede ver la alimentación del PLC y como se conectan las entradas y las salidas del PLC.

Página 3: Se definen las conexiones de los puertos Ethernet. Todos los dispositivos se conectan con el switch. El switch se conecta con un puerto que tiene conexión a internet.

Página 4-5: Planos con la ubicación de los elementos del panel con la botonera y los indicadores, así como sus conectores. Cada selector activa una entrada en el PLC, mientras que los indicadores se activan mediante salidas del PLC.

Página 6: Muestra la placa del cuadro y la ubicación de las canaletas y las guías. Se utiliza para cortar y ubicar correctamente las piezas.

Página 7: Representación de las bornas y la numeración.

Página 8: Posición de todos los elementos en el cuadro eléctrico. Facilita la información de cada elemento y donde montarlo.

3.5.3 Instalación

Una vez se ha completado el diseño del cuadro eléctrico con la ubicación de los elementos eléctricos y los dispositivos, se monta el mismo. A continuación, las figuras 20 a 25 muestran las fases del montaje.

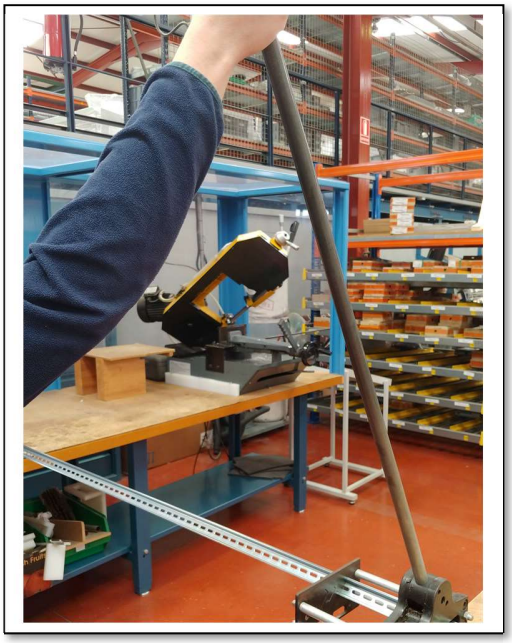


Figura 20 y 21: Cortar Canaletas y perfiles

Figura 22: Montaje de canaletas y perfiles Din.

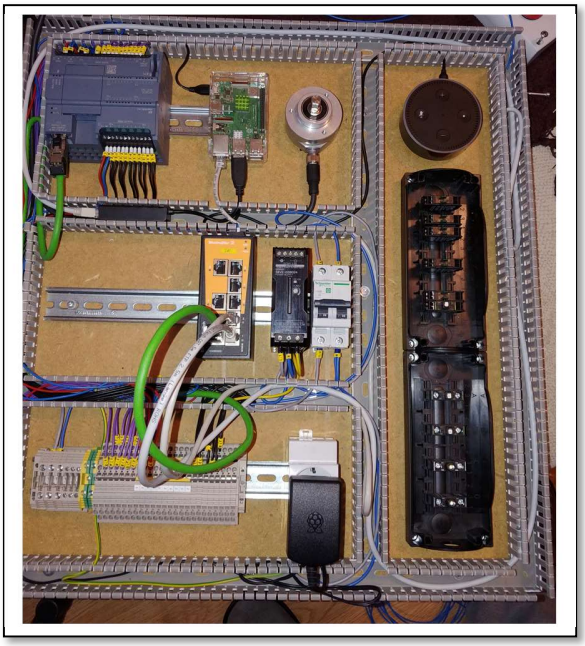
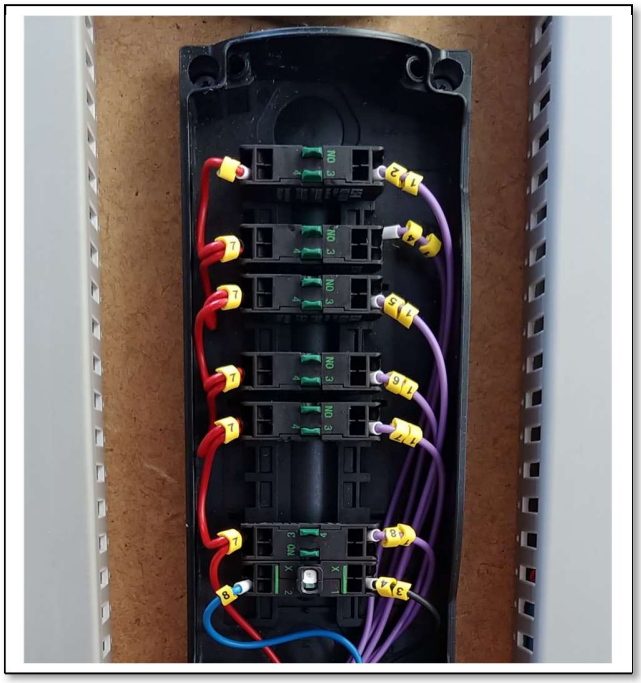


Figura 23: Montaje de los dispositivos y elementos eléctricos.



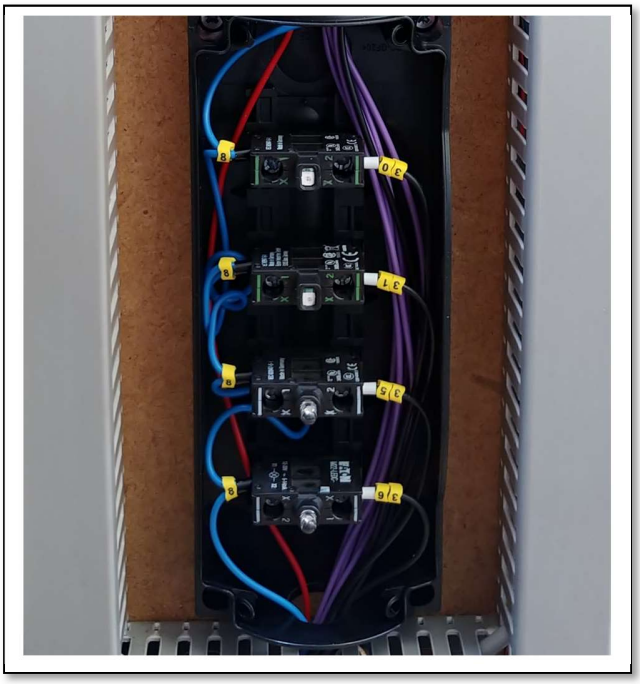
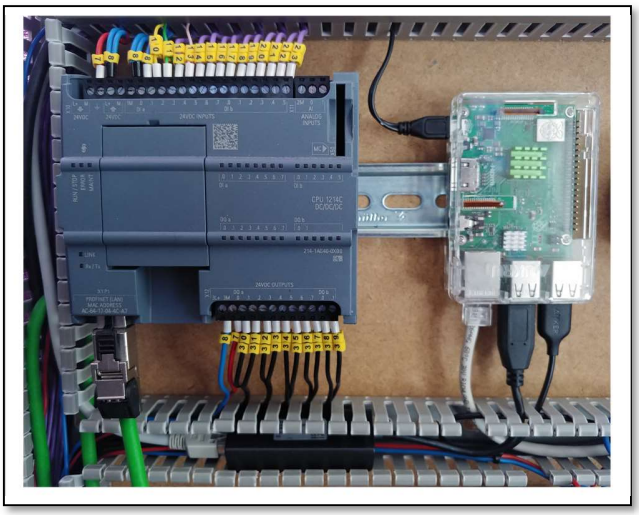


Figura 24 y 25: Montaje de las botoneras.



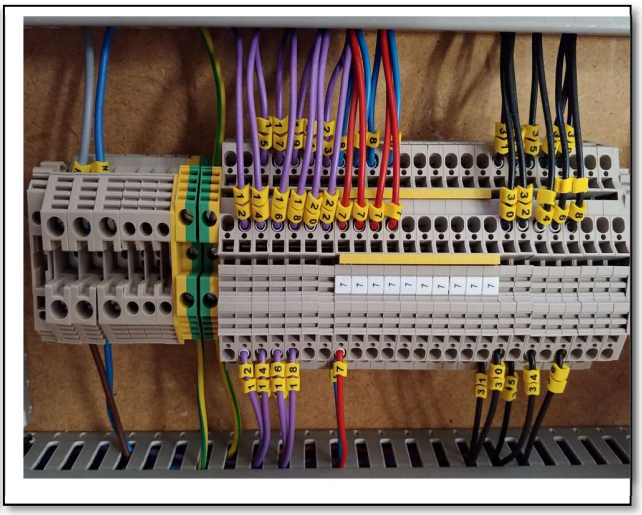


Figura 26 y 27: Montaje de maniobra y PLC.

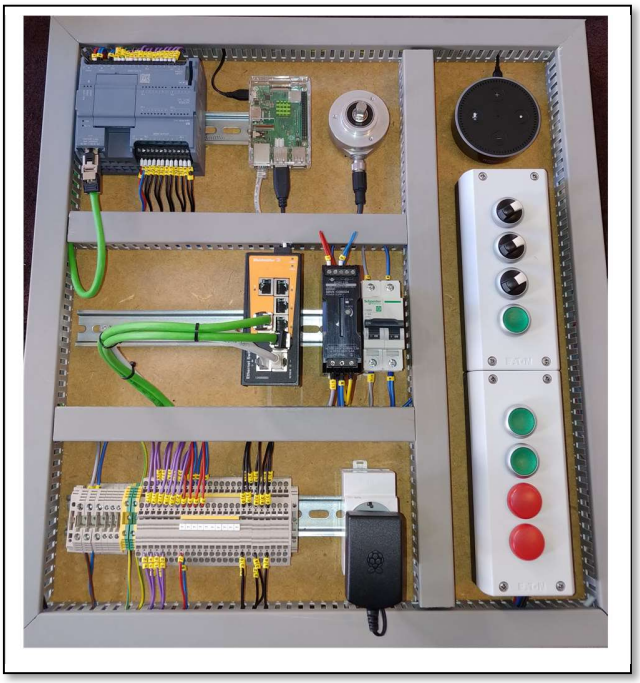


Figura 28: Cuadro Eléctrico.

El resultado final se puede apreciar en la figura 28. Como se puede ver, en las diferentes fases de la instalación del cuadro eléctrico, éste es un elemento sencillo pero complicado a la vez. El usuario puede interactuar de dos distintas formas: activar las entradas de PLC a través de los botones del panel, como

si estuviese delante de una máquina, o a través de Amazon Alexa con el altavoz Echo, interactuando con la interfaz de voz desarrollada.

4. Pruebas

4.1 Comunicación entre PLC y Raspberry Pi

A continuación, se muestran las pruebas de comunicación entre el PLC y la Raspberry Pi, utilizando el protocolo Modbus TCP/IP.

En la Raspberry Pi se configura el adaptador de USB a Ethernet. La dirección IP que se utiliza es 192.168.2.X. Es una dirección fija que el PLC y la Raspberry Pi utilizan para comunicar en el mismo rango.

Primero se ajusta la dirección IP. Para configurar la dirección en el adaptador se selecciona “Wireless & Wired Networks Settings” en el escritorio de la Raspberry Pi. Está ubicado en la esquina superior a la derecha, en el icono de las redes actuales. Aparece una ventana con el nombre “Network Preferences”. Se configura el “interface eth1” y en IPv4 se asigna la dirección 192.168.2.15 y DNS Server 192.168.2.15. El resto se deja sin añadir nada más.

Para comprobar que la comunicación entre ellos es correcta se ejecuta el comando "ping" en una ventana de consola. Se utiliza para comprobar que existe comunicación entre dos equipos una red a nivel de protocolo IP.

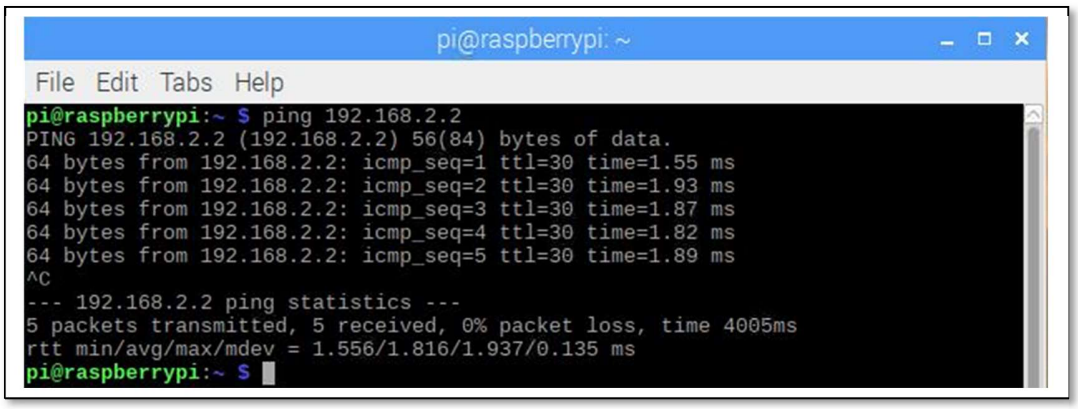


Figura 29: “ping” entre PLC y Raspberry Pi

Por último, se utiliza la librería Pymodbus para intercambiar datos entre la Raspberry Pi y el PLC. Se utiliza un “script” que se ejecuta solo una vez y activa una salida de PLC. Una vez que se ha ejecutado el “script” la salida 1 debe estar activada. De esta forma se comprueba que la comunicación con el protocolo MODBUS TCP/IP es correcta y está funcionando.

```
from pymodbus.client.sync import ModbusTcpClient as ModbusClient
con = ModbusClient(host= '192.168.2.2')
def test():
    con = ModbusClient(host= '192.168.2.2')
    con.write_coil(0,1)
    con.close()
test()
```

4.2 Comunicación entre Alexa y Raspberry Pi

Se muestra a continuación las pruebas realizadas entre el Alexa y Raspberry Pi, entre Alexa y la Raspberry Pi, utilizando HTTPS y Ngrok.

Después de instalar el programa Ngrok en la Raspberry Pi, se ejecuta el programa en el directorio donde se ha instalado. Así, si se ejecuta el comando "sudo ./ngrok http 5000", aparece en la línea de comandos la URL de HTTPS que se utiliza. En la figura 30 se muestra un ejemplo donde se puede ver la URL, la región y varios datos más. Es importante tener en cuenta que, en la versión gratuita, la dirección HTTPS cambia en cada nueva sesión.

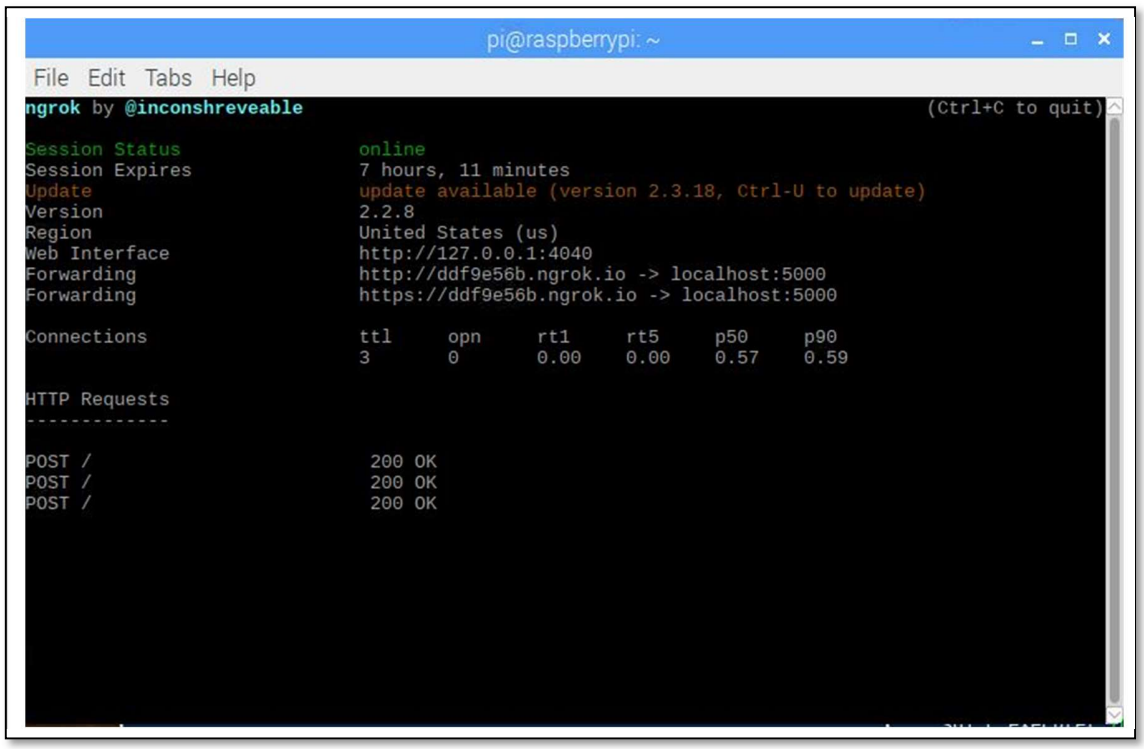


Figura 30: Prueba de Ngrok en la Raspberry Pi.

En la plataforma de Alexa se copia la dirección de HTTPS y se guarda. Seleccionando Test en la parte superior se puede comprobar que el código de Alexa y el código en la Raspberry Pi están funcionando y comunicando a través del túnel que ha creado Ngrok.

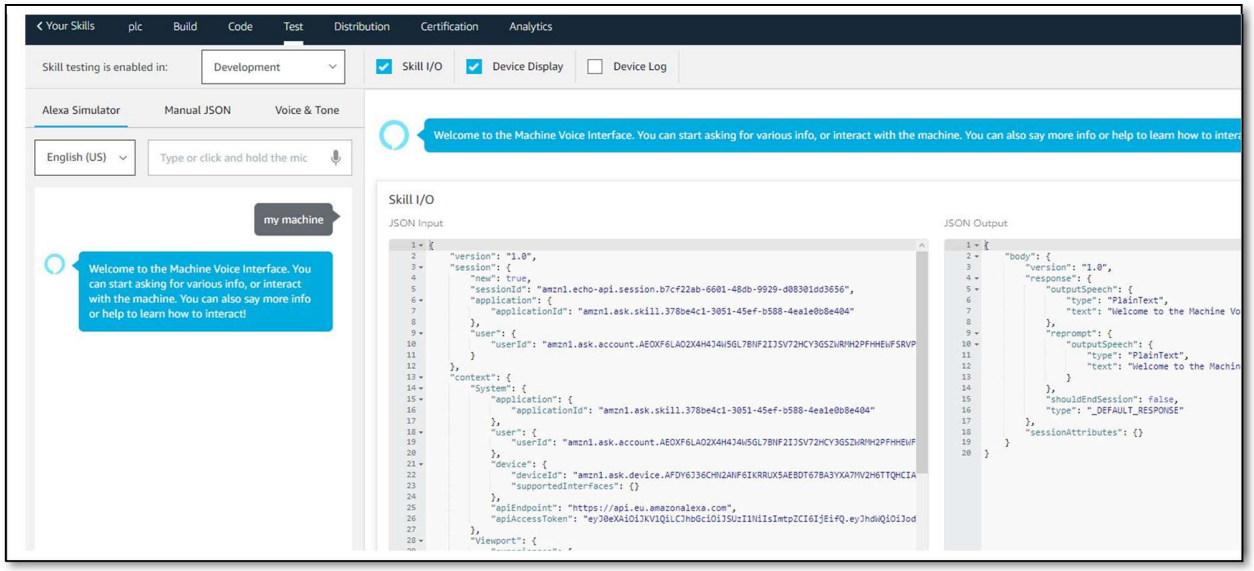


Figura 31: Alexa y Raspberry Pi.

Si el “skill” no tiene una respuesta de la Raspberry Pi o el código del “intent” tiene un error, la consola de pruebas pone que tipo de error es.

Si por ejemplo Alexa no recibe una respuesta de la Raspberry Pi el error que se obtiene es el siguiente:

"request": {

"type": "SessionEndedRequest",

"requestId": "amzn1.echo-api.request.37aef208-6cc0-403d-8089-ad2ee02e0d41",

"timestamp": "2019-03-24T17:03:31Z",

"locale": "en-US",

"reason": "ERROR",

"error": {

"type": "INVALID_RESPONSE",

"message": "An exception occurred while dispatching the request to the skill."}

El sistema funciona correctamente en todas las pruebas realizadas. Es importante tener en cuenta que la respuesta del sistema depende de la calidad de conexión de internet y y del tiempo de retraso en la comunicación entre el sistema con altavoz Echo y la Raspberry Pi, y los servidores de Amazon.

4.3 Pruebas con Amazon Echo y el cuadro eléctrico

Se muestra a continuación las pruebas realizadas entre el Amazon Echo y Raspberry Pi, utilizando el altavoz Echo y las botoneras del cuadro.

El modo de control con el cuadro eléctrico es el modo estándar, como si el usuario estuviese delante de la máquina. Las pruebas consisten en activar y desactivar las salidas a través del panel de pulsadores y comprobar que el código funciona correctamente. También se prueba que el altavoz Echo puede reconocer correctamente al usuario. La primera prueba es activar las entradas y las salidas del PLC. En la figura 32 se activa la entrada 2 del PLC.

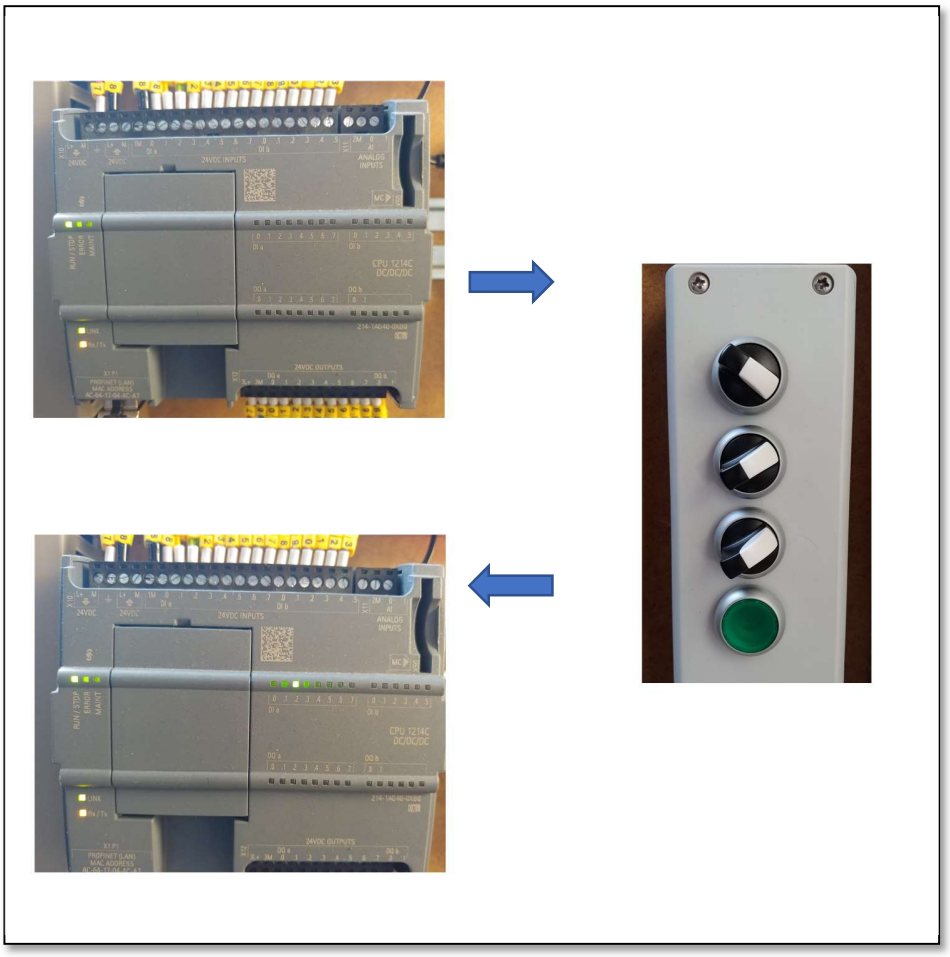


Figura 32: Pruebas de pulsación de botones.

La figura 33 muestra cómo se activan las salidas 4 y 5, y los indicadores correspondientes, al actuar sobre los mandos conectados a las entradas y salidas del PLC.

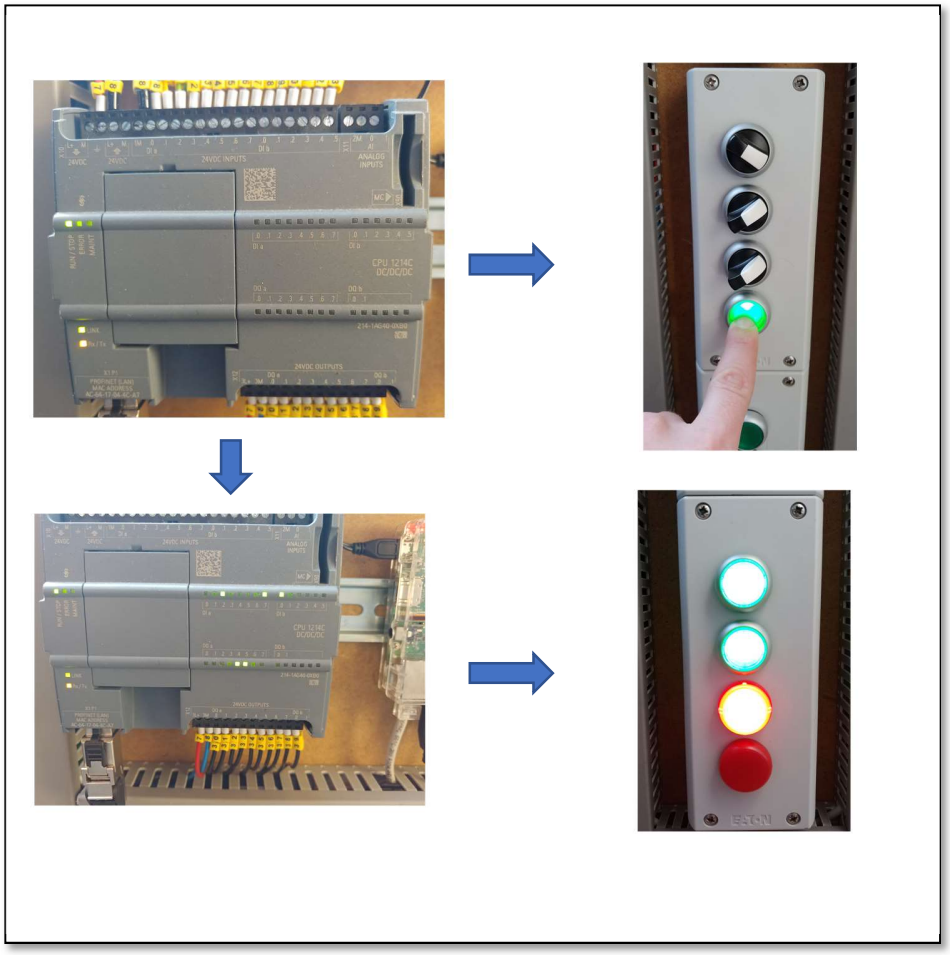


Figura 33: Pruebas de la botonera y las entradas, salidas del PLC.

El programa del PLC funciona del siguiente modo. Cuando se pulsa el botón verde que activa la entrada 8, el PLC comprueba si la entrada 7 está activada. Si la entrada esta activada, considera que esta es modo “standby”, esto permite poner la máquina en funcionamiento automático. Entonces la salida 5 se activa, y el primer LED rojo en el panel de pulsadores e indicadores se enciende, indicando el funcionamiento de la máquina.

La segunda prueba es entre el usuario y el altavoz Amazon Echo. Cuando el usuario utiliza la frase “**Echo**” se despierta el altavoz y está en espera para la frase “My machine”. Utilizando la frase “My machine” se conecta con el “skill” desarrollado. En la figura 34 se muestra la prueba con el altavoz Amazon Echo y la plataforma Alexa en condiciones reales.

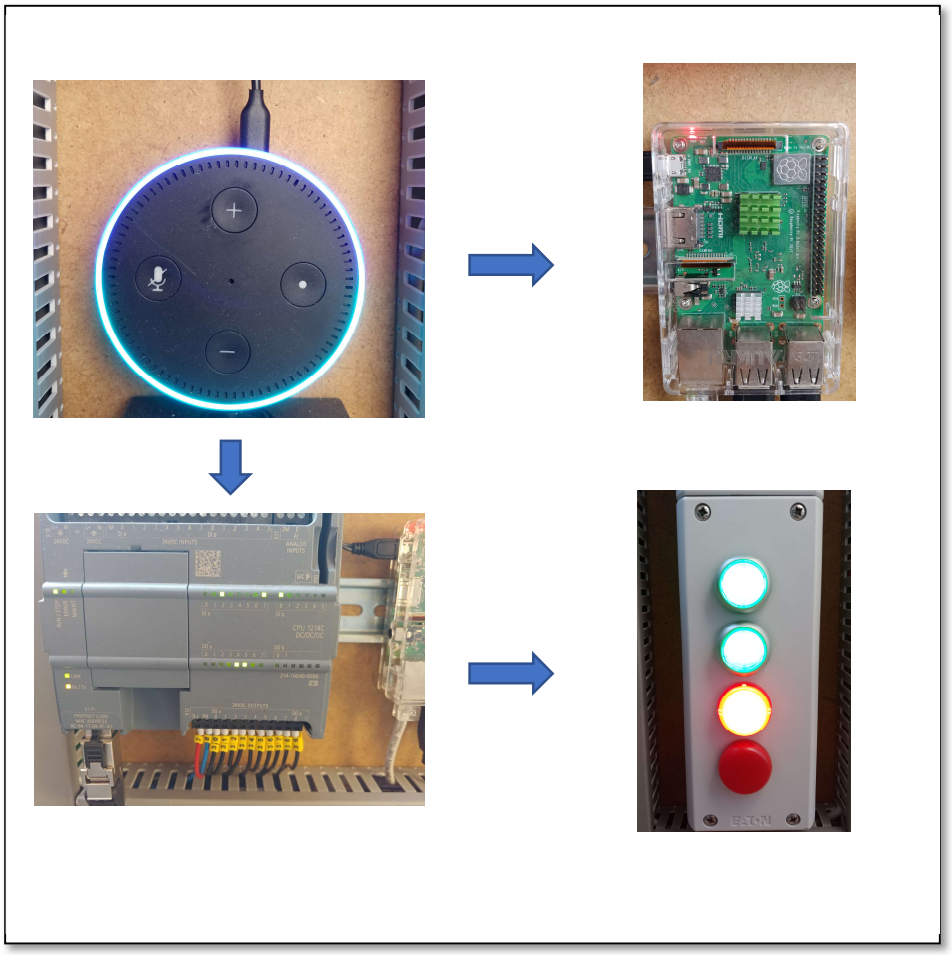


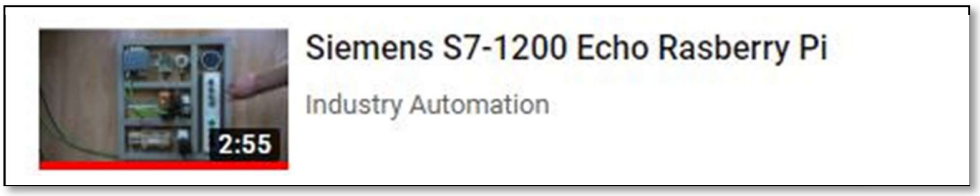
Figura 34: Pruebas de Amazon Echo.

En esta prueba el usuario utiliza la frase “Activate the output 3” después de iniciar el "skill". En este momento se conecta con la Raspberry Pi, y ésta ejecuta el código relacionado con el “intent” que el usuario acaba de activar, encendiéndose el piloto conectado a la salida 3 del PLC.

El Cuadro funciona correctamente en todas las pruebas realizadas, tanto con el panel del cuadro y la botonera como con de Alexa. Como se aprecia en las diferentes pruebas realizadas todos los sistemas tienen un respuesta rápida y correcta. El usuario puede interactuar sin problema con el panel de pulsadores e indicadores y con el Echo de Amazon.

4.4 Video

Se ha grabado un vídeo con diferentes pruebas realizadas sobre el sistema completo, en el cual se aprecia mejor el funcionamiento del sistema, y como responde ante acciones de un usuario sobre el panel de botones y la interfaz de voz para Alexa. El vídeo está disponible en el siguiente enlace:



https://www.youtube.com/watch?v=QTJ21SYIMio&index=8&list=PLRcqyyEn618I2GEmRjtvRzo1CiMNSzxZ2&t=0s

5. Conclusiones

Como conclusión principal cabe mencionar que se ha podido realizar un sistema que dispone de una interfaz humano-máquina de voz, además de una interfaz estándar a través de un panel de botones. El sistema desarrollado es multiplataforma y funciona con cualquier PLC, siempre que éste tenga disponibles un puerto Ethernet y el protocolo estándar Modbus TCP/IP. También se consideró inicialmente protocolos propietarios de la marca Siemens, como es Step7, pero estos protocolos son muy específicos, y limitan la funcionalidad del sistema, además de no poderlo utilizar con PLCs de otras macas en el futuro. De la forma en que se ha desarrollado, es posible utilizar el sistema desarrollado en cualquier aplicación y cualquier PLC, y solo es necesario cambiar las direcciones IP que utiliza el "script" de Modbus en la Raspberry Pi.

Se ha comprobado que el usuario puede ejecutar el "skill" de Alexa, usando el altavoz de Amazon Echo, correctamente sin problemas. La interfaz desarrollada en la plataforma de Amazon permite tomar el control de la aplicación y consultar datos complejos de manera bastante intuitiva, sin la necesidad de interactuar con un panel HMI o una pantalla. La comunicación entre el PLC y la Raspberry Pi se ejecuta automáticamente en intervalos predefinidos, sin ninguna intervención del usuario, y los datos que la Raspberry Pi captura del PLC se almacenan en la base de datos SQL-lite. Esto es una gran ventaja porque permite el almacenamiento automático de los datos siempre que la aplicación está funcionando, aunque no haya conexión a internet. Así, aunque no haya conexión a internet la base de datos sigue recibiendo datos, permitiendo la consulta posterior de estos datos a través del altavoz Echo cuando se restablece el acceso a internet.

El cuadro eléctrico diseñado ha permitido validar el sistema con una aplicación de ejemplo, sencilla pero real, con la que es posible controlar las entradas y salidas del PLC, además de examinar la interacción entre todos los elementos en tiempo real. El sistema montado tiene una respuesta bastante rápida, y esta respuesta permite al usuario final interactuar con el altavoz Echo de manera natural, como una conversación real entre dos personas. El manejo de todas las peticiones de Alexa se hace locamente en la Raspberry Pi, utilizando Python y varias librerías. Usar el lenguaje Python ha permitido programar muy rápidamente y de manera eficaz.

El resultado final es una maqueta que simula bien una aplicación real sencilla. Pero esta aplicación podría perfectamente ser más compleja, como el control de una máquina o una vivienda, y en ambos casos el control se haría a través de la interfaz de Voz. Es importante tener en cuenta el lugar donde se ubica el cuadro eléctrico y el altavoz Echo. En el caso hipotético de que el cuadro eléctrico con la Raspberry Pi formara parte de una máquina instalada en una fábrica o en un sitio donde el ruido en el ambiente fuese alto, sería normal que el altavoz Echo no diera los resultados deseados.

La conclusión final del proyecto, es que permite aumentar la capacidad de conectividad de cualquier aplicación de automatización basada en un PLC, con un coste muy bajo. Además, cuando se utiliza una Raspberry Pi en sistema de automatización, se pude sacar provecho de la versatilidad y estabilidad un sistema operativo como Raspbian.

5.1 Proyecto Futuro

En este apartado se exponen posibles desarrollos futuros sobre el proyecto desarrollado, mostrando así también la capacidad que este proyecto tiene de ser ampliado. Estos posibles desarrollos futuros se consideran a partir de la base ya definida en este proyecto.

- Desarrollar una aplicación para Android.

  Desarrollar una aplicación para Android que permita acceder la Raspberry Pi localmente. Se podría utilizar Wi-Fi para acceder la Raspberry Pi. Esto implica un nuevo desarrollo y añadir más protocolos de comunicación en la Raspberry Pi. La aplicación para Android se podría escribir en lenguajes como Kotlin, Java y C ++, usando el kit de desarrollo de software de Android, mientras que también es posible usar otros idiomas. También habría que desarrollar una nueva interfaz para el acceso al sistema desde el móvil.
- Utilizar MQTT (Message Queue Telemetry Transport).

  Se podría programar la aplicación de Android y la Raspberry Pi utilizando el protocolo de comunicación MQTT. MQTT es un protocolo de conectividad de máquina a máquina. Fue diseñado como un transporte de mensajes de publicación y suscripción extremadamente ligero. Es útil para las conexiones con ubicaciones remotas donde se requiere que el código sea ligero y el ancho de banda de la red no es estable. Por ejemplo, se ha utilizado con sensores que se comunican con un intermediario a través de un enlace satelital, a través de conexiones telefónicas y en una variedad de escenarios de dispositivos pequeños y automatización del hogar. También es ideal para aplicaciones móviles debido a su pequeño tamaño, bajo consumo de energía, paquetes de datos minimizados y distribución eficiente de información a uno o varios receptores.
- Utilizar una Raspberry Pi industrial.

  Finalmente, proyecto se ha abordado con una Raspberry Pi estándar, principalmente debido a la ventaja de su bajo coste. La desventaja es que el entorno industrial es un entorno que requiere otro tipo de electrónica, más robusta. En el futuro se podría utilizar una Raspberry Pi que cumpliría los estándares de industria. En este momento existen varias opciones que se podrían utilizar, aunque algunas de ellas necesitarían adaptar un poco el código en la Raspberry Pi. El coste sería mucho más elevado en comparación con una Raspberry Pi estándar del mercado.

- Utilizar otro tipo de reconocimiento de voz.

  Probar a utilizar otros tipos de sistemas de reconocimiento de voz. Configurar la aplicación y documentar los resultados. Posibles opciones pueden ser Google Voice, Siri o Mycroft, siendo esta última una opción "open source".

- Utilizar otras marcas de PLC.

  Se puede abordar la modificación del sistema para usar PLCs de otras marcas populares y extendidas, como pueden ser Allen-Bradley, Schneider u Omron. Habría que configurar el protocolo Modbus para los PLCs escogidos, y asignar las variables que deben ser accesibles desde la Raspberry Pi. Debería ser una tarea sencilla, siempre que se utilice un PLC que soporte Modbus sobre Etnernet y TCP/IP.

- Conectar varios PLCs con la Raspberry.

  Una ampliación interesante del proyecto sería comunicar la Raspberry Pi con varios PLCs al mismo tiempo, y analizar los resultados.

6. Anexos

Anexo 1. Esquemas eléctricos y de montaje

Página 1: Alimentación Cuadro.

Página 2: PLC Maniobra.

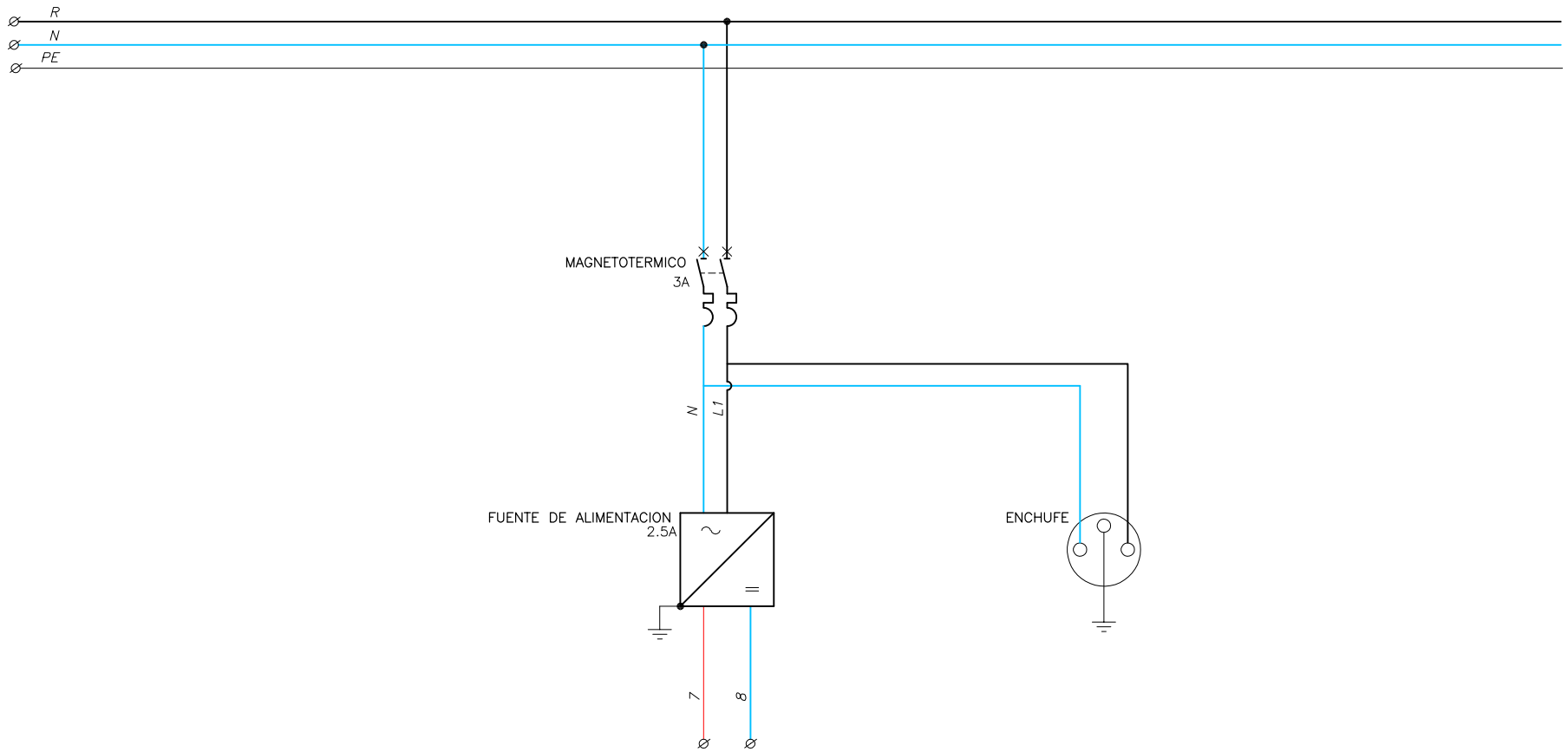
Página 3: Conexiones.

Página 4-5: Panel de pulsadores e indicadores.

Página 6: Cuadro Canaletas y Guías.

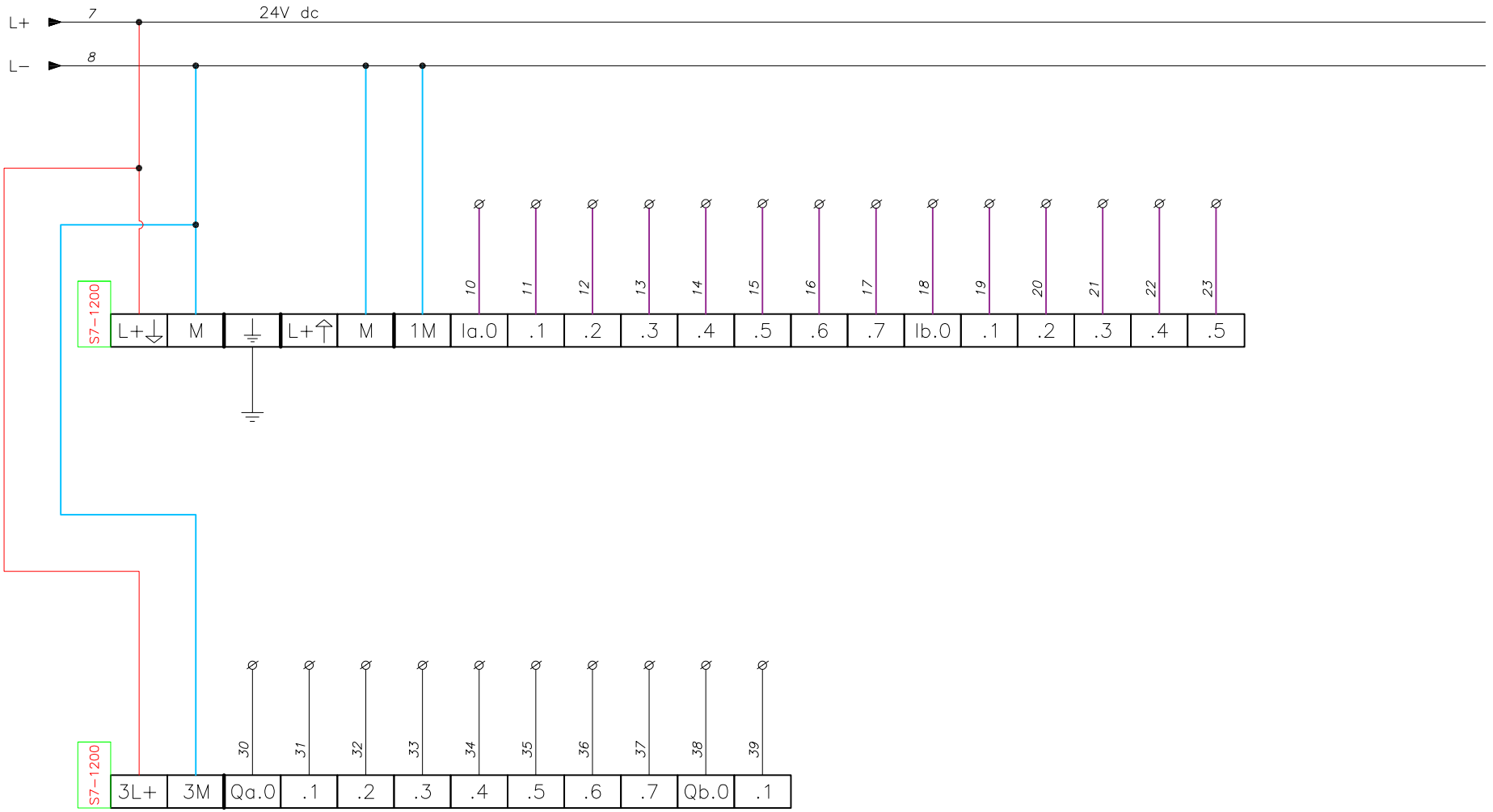
Página 7: Bornas.

Página 8: Posición Elementos.

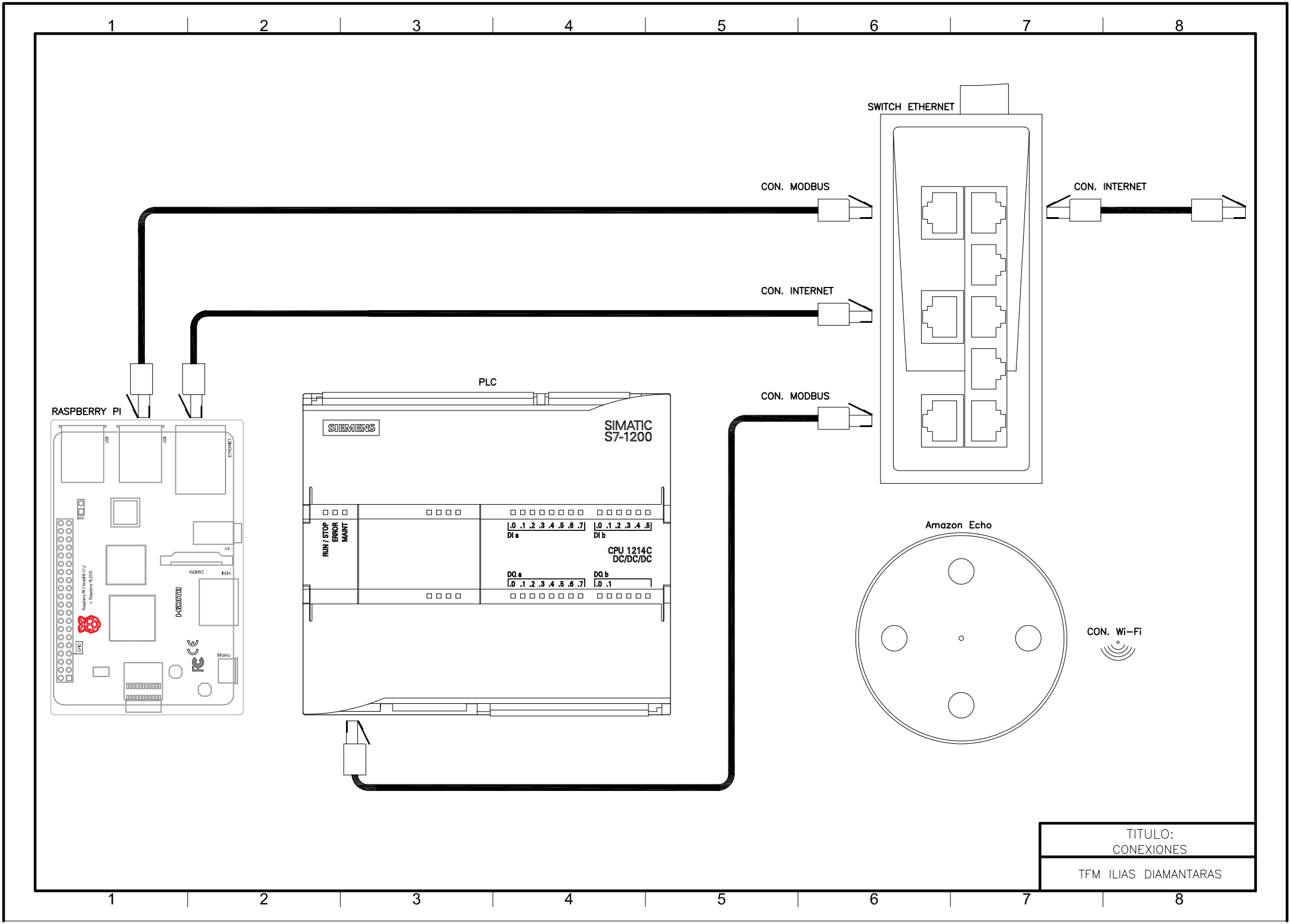


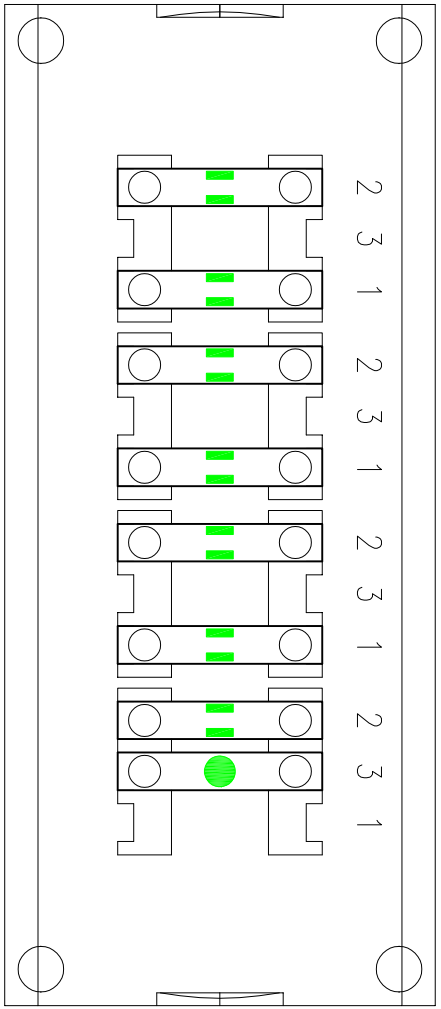
TITULO:
ALIMENTACION CUADRO

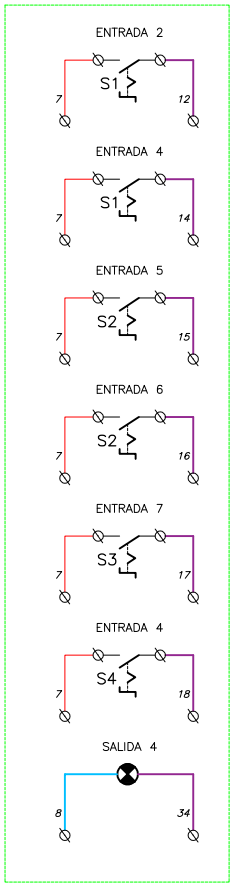
TFM ILIAS DIAMANTARAS

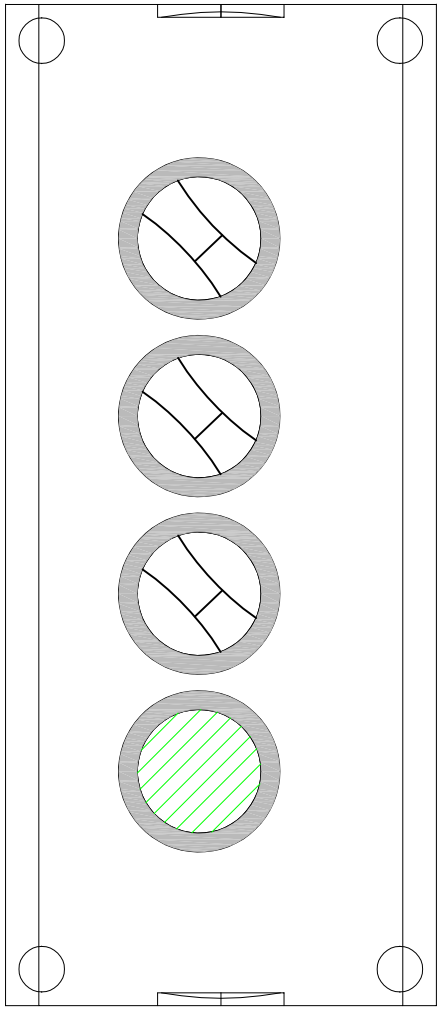


TITULO: PLC MANIOBRA
TFM ILIAS DIAMANTARAS


SWITCH ETHERNET
CON. MODBUS
CON. INTERNET
CON. INTERNET
CON. MODBUS
RASPBERRY PI
PLC
SIEMENS
SIMATIC
S7-1200
RUN / STOP
ERROR
MAINT
DI a
DI b
CPU 1214C
DC/DC/DC
DQ a
DQ b
Amazon Echo
CON. Wi-Fi
TITULO:
CONEXIONES
TFM ILIAS DIAMANTARAS

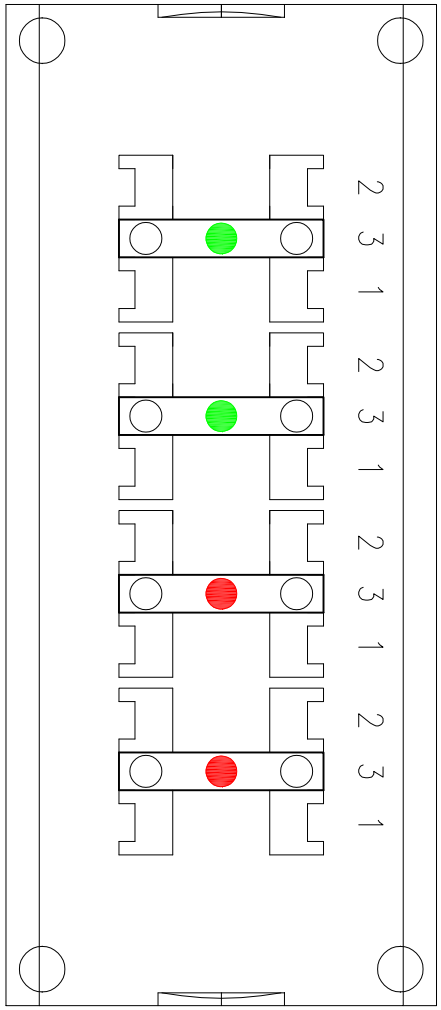


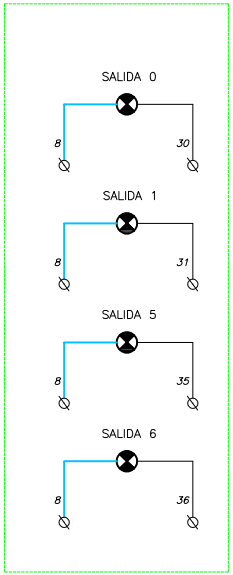


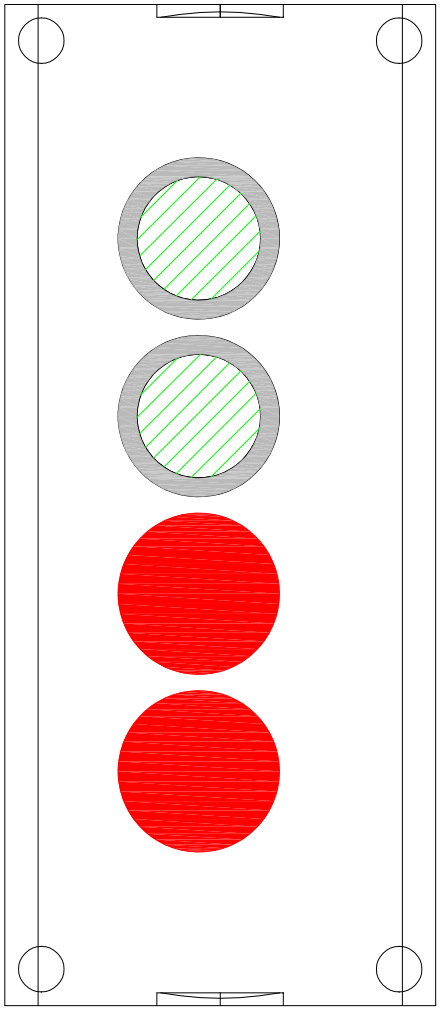


TITULO:
PULSADORE E INDICADORES 1

TFM ILIAS DIAMANTARAS

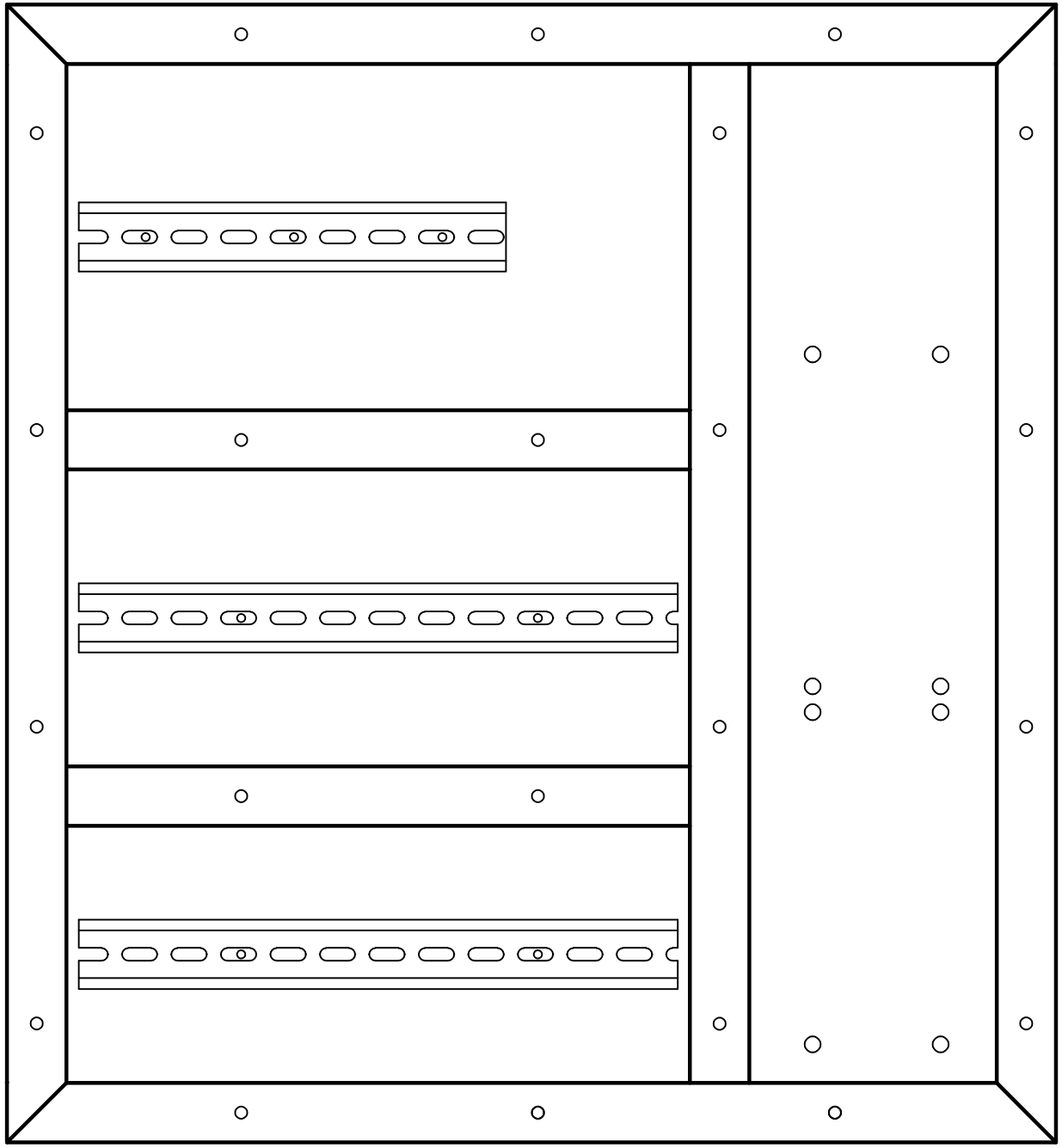






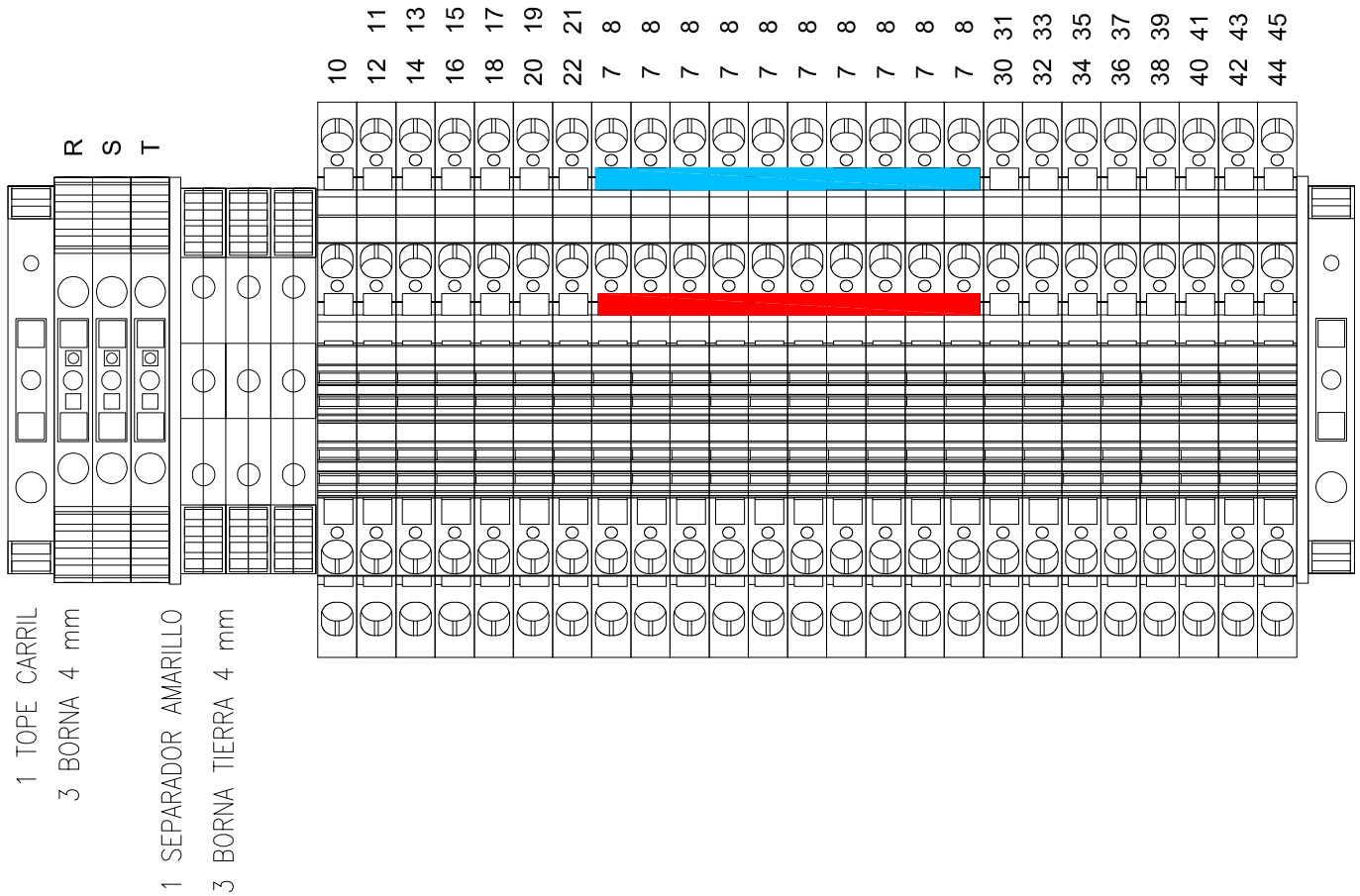
TITULO:
PULSADORE E INDICADORES 2

TFM ILIAS DIAMANTARAS



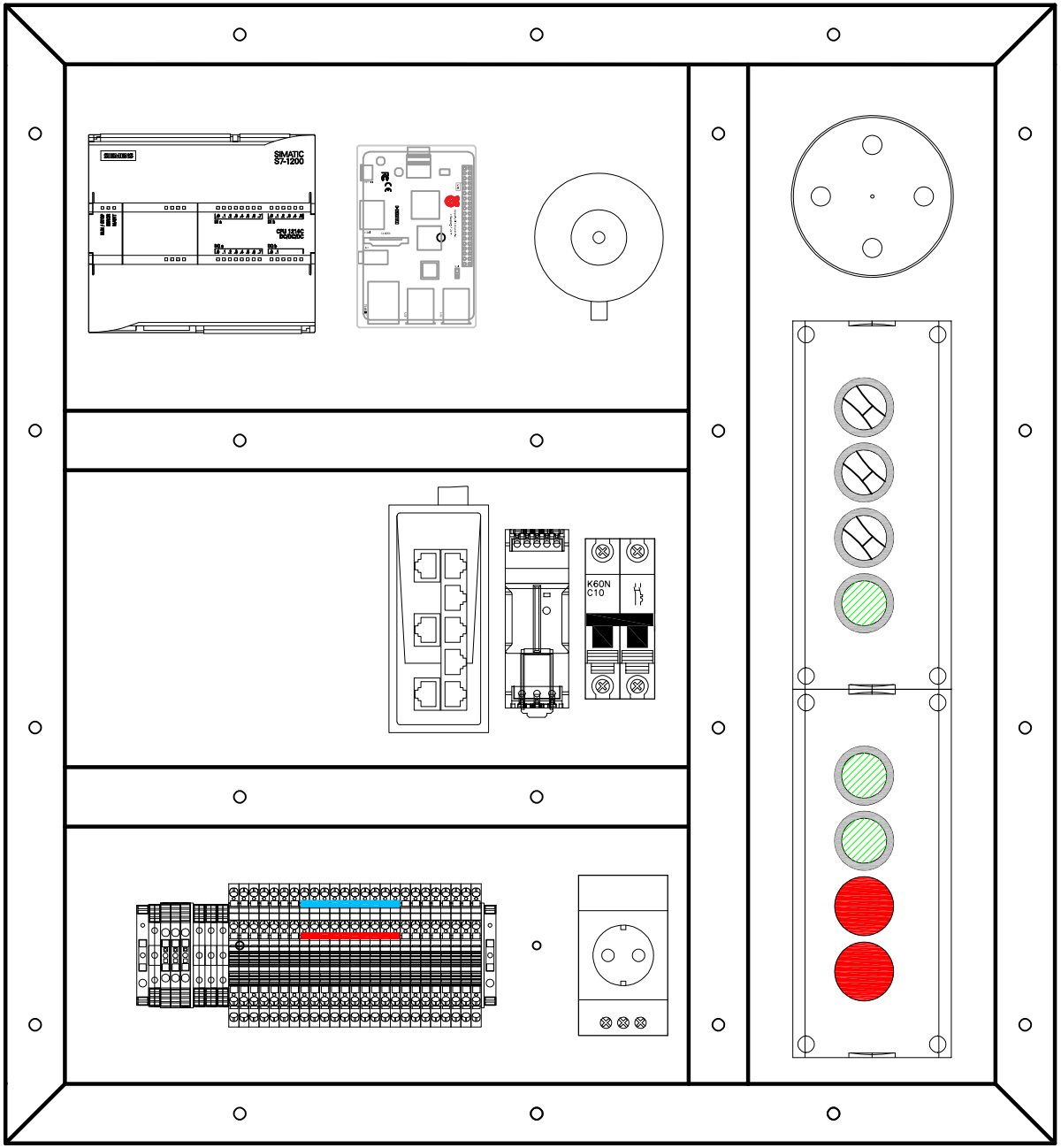
TITULO:
CUADRO CANALETAS Y GUIAS

TFM ILIAS DIAMANTARAS



TITULO:
BORNAS

TFM ILIAS DIAMANTARAS



TITULO:
POSICION ELEMENTOS

TFM ILIAS DIAMANTARAS

ANEXO 2 BIBLIOGRAFIA

[1] Autodesk overview. Consultado 2019, Abril. *Autocad* Retrieved from Autodesk: https://www.autodesk.com/products/autocad/overview

[2] AWS Lambda, Amazon aws website. Consultado en abril de 2019:

https://aws.amazon.com/es/lambda/features/

[3] Benchmark Tests for the ARPA Spoken Language Program, Proc. Of the 1995 ARPA Human

Language Technology Workshop, pp. 5-36, 1995

[4] Diseño e Interacción del habla. Christine Murad, Leigh Clark 2018. Consultado en abril de 2019:

https://dlnext.acm.org/doi/abs/10.1145/3236112.3236149

[5] Build your first app, Google apps. Consultado en abril de 2019:

https://developer.android.com/training/basics/firstapp/

[6] Información de Endpoint, Socrata. Consultado en abril de 2019:

https://dev.socrata.com/docs/endpoints.html

[7] Amazon Alexa. Amazon developer. Consultado en abril de 2019:

https://developer.amazon.com/alexa

[8] Flask microframe work. Flask descripción. Consultado en abril de 2019:

https://pypi.org/project/Flask/

[9] Flask-Ask. Flask-Asl. Consultado en abril de 2019:

https://flask-ask.readthedocs.io/en/latest/index.html

[10] Voice Recognition "FGC recognicion server". Fifth Generation Computer Corporation 2007

Consultado en abril de 2019:

http://www.fifthgen.com/speaker-independent-connected-s-r.htm

[11] Protocolo Pymodbus. Github Python Modbus stack. Consultado en abril de 2019:

https://github.com/riptideio/pymodbus

[12] TCP/IP Protocol. IBM knowledge Center. Consultado en abril de 2019:

https://www.ibm.com/support/knowledgecenter/en/ssw_aix_71/com.ibm.aix.networkcomm/tcpip_protocols.htm

[13] Automation History. Jim Pinto, 2007. Consultado en abril de 2019:

http://www.jimpinto.com/writings/automationhistory.html

[14] Protocolo Modbus. Modbus Organization. Consultado en abril de 2019:

http://www.modbus.org

[15] Voice Interfaces: Assessing the Potential. Jakob Nielsen, Nielsen Norman Group, 2003. Consultado en abril de 2019: https://www.nngroup.com/articles/voice-interfaces-assessing-the-potential/

[16] What is Ngrok. Ngrok product. Consultado en abril de 2019: https://ngrok.com/product

[17] Siemens TIA Portal. Siemens automation software. Consultado en abril de 2019: https://new.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal.html

[18] Sistema operativo Raspbian. Raspbian Organization. Consulatdo en abril de 2019. https://www.raspbian.org

[19] Kunbus Industrial Raspberry Pi. Kunbus Pi series. Consultado en abril de 2019: https://revolution.kunbus.com/

[20] About Sqlite. Sqlite about webpage. Consultado en abril de 2019: https://sqlite.org/about.html

[21] Sistemas de automatización y autómatas programables. Enrique Mandado Pérez y otros. Marcombo, 2018.

[22] Automation Speech Recognition-A Brief History. B.H. Juang, Lawrence R. Rabiner 2004. Consultado en abril de 2019: https://web.archive.org/web/20140817193243/http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/354_LALI-ASRHistory-final-10-8.pdf

[23] The rise of voice timeline. Witlingo company. Consultado en abril de 2019: http://www.witlingo.com/the-rise-of-voice-timeline/

[24] Shoebox IBM history. IBM Exhibits. Consultado en abril de 2019: https://www.ibm.com/ibm/history/exhibits/specialprod1/specialprod1_7.html